RL-TR-92-277
Final Technical Report
December 1992

AIR DEFENSE INITIATIVE SIMULATION FOR COMMAND AND CONTROL DEVELOPMENT (ADISC2)

Martin-Marietta Corporation

Lonnie Fouty

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

Rome Laboratory
Air Force Materiel Command
Griffiss Air Force Base, New York

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RL-TR-92-277 has been reviewed and is approved for publication.

APPROVED:

JERRY F. LIPA
Project Engineer

FOR THE COMMANDER: John A Graniero

JOHN A. GRANIERO
Chief Scientist
Command, Control and Communications Directorate

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify RL(C3AA) Griffiss AFB, NY 13441-4505. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY (Leave Blank)

2. REPORT DATE
December 1992

3. REPORT TYPE AND DATES COVERED
Final Sep 87 - Feb 92

4. TITLE AND SUBTITLE
AIR DEFENSE INITIATIVE SIMULATION FOR COMMAND AND CONTROL DEVELOPMENT (ADISC2)

5. FUNDING NUMBERS
C - F30602-87-C-0221
PE - 63741D
PR - 3379
TA - 01
WU - 01

6. AUTHOR(S)
Lonnie Fouty

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Martin-Marietta Corporation
Information and Communications Division
P.O. Box 1260
Denver CO 80201-1260

8. PERFORMING ORGANIZATION REPORT NUMBER
ADI-02-B011-000

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Rome Laboratory (C3AA)
525 Brooks Road
Griffiss AFB NY 13441-4505

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
RL-TR-92-277

11. SUPPLEMENTARY NOTES
Rome Laboratory Project Engineer: Jerry F. Lipa/C3AA/(315)330-4494

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The objective of this effort was to develop an integrated simulation to provide threat, surveillance, engagement, and communications models which could be exercised in accordance with threat scenarios to provide a consistent baseline with which to develop technology to demonstrate man-in-the-loop C^2 functions in support of the Air Defense Initiative. This was achieved in the simulation which utilizes the National Testbed (NTB) simulation executive as legacy software to control the simulation during execution. This allows execution of a wide range of scenarios, reducing the integration of simulated functions to a generic message-passing activity between the simulation nodes. The simulation support environment provides services to the user during pre-simulation, runtime, and post-processing modes. The simulation environment is comprised of several integrated Commercial-Off-The-Shelf products such as INGRES for relational database management, FIGARO for animation and graphics, Sun-tools for window management, Template/BLOX for user interface management, and NAG for statistical analysis.

14. SUBJECT TERMS
Simulation, Air Defense Initiative.

15 NUMBER OF PAGES
92

16 PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Summary

The primary objective of the contracted Air Defense Initiative Simulation for Command and Control (ADISC^2) Development effort was to design, implement, install, test, operate and maintain an integrated simulation to provide Threat, Surveillance, Engagement, and Communications models which may be exercised in accordance with Government-provided threat scenarios. All objectives were achieved or exceeded.

The long-term objective of the ADISC^2 is to provide a consistent baseline to develop the technology to demonstrate man-in-the-loop C^2 functions in support of the Air Defense Initiative (ADI).

To meet the objectives of the ADISC^2 program, Martin Marietta Information & Communications Systems, under contract to RADC/COAA developed a simulation capability providing a consistent baseline for development and evaluation of potential applications of command and control (C^2) technology for strategic air defense requirements. This development incorporated the functional modules of **Threats** (bombers, air-launched cruise missiles, gravity bombs, submarines, sub-launched cruise missiles, electronic jammers and tankers), **Sensors** (ie., JSS, NWS, Alaskan radar, OTH-B, SEEK SKYHOOK, Space-Based Radar, SOSUS, SURTASS, and airborne and shipborne radars, to name a few), **Engagement** elements (F-15s, SAMs, etc.), and representation of **Environment,** and **Communications**. An **Air Traffic** Module is provided to allow realistic simulation of background activities (commercial, military airlift, and Strategic Air Command) in support of air defense engagement scenarios. Figure 1 (on the following page) presents the functional relationship among these elements.

The ADISC^2 incorporates the National Test Bed (NTB) Simulation Executive as legacy software used to control the simulation during execution. The use of this Executive allows execution of a wide range of scenarios, reducing the integration of simulated functions to a generic message-passing activity between nodes of a distributed network.

The air defense applications are supported by a Simulation Support Environment (SSE) that provides services to the user during Pre-Simulation, Runtime and Post-processing modes. The SSE is comprised of several integrated Commercial-Off-The-Shelf (COTS) products such as INGRES for relational database management, FIGARO for animation and graphics, Sun-tools for window management, Template/Blox for user interface management, and NAG for statistical analysis. This powerful composite provides a highly flexible and extensible environment that allows the user to rapidly configure simulations to a wide range of operational evaluation and analytical requirements.

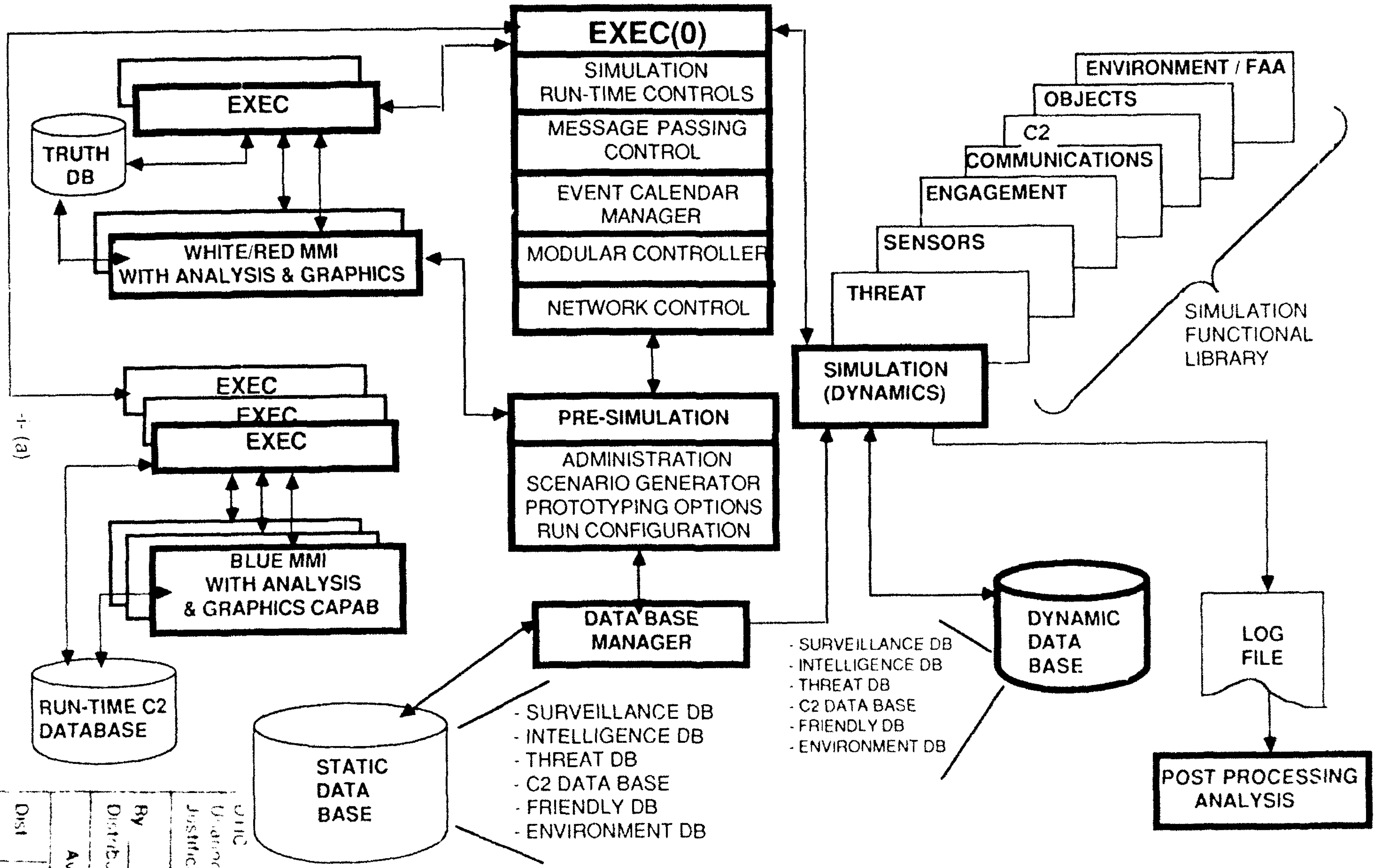
This flexibility is made possible, in part, by the object-oriented concepts employed in the design of the SSE. The design of ADISC^2 utilizes standard object-oriented concepts and definitions but breaks with certain methods employed in most widely available COTS applications. This has also produced a slightly different vocabulary than is normally associated with object-oriented analysis and design.

Tables may be created to incorporate and structure sets of attributes that describe objects. Tables incorporate attributes that uniquely identify objects, establish performance envelopes for objects, define the operational or mission environment of objects, or establish the basis for potential action/interaction among objects, systems, sub-systems, or components.

The logical layering used in the ADISC^2 design for object definition is:

- Structural Frames within an environmental medium
- Category
- Class
- Object

FIGURE 1. - ADISC2 DYNAMIC SIMULATION FUNCTIONAL DESIGN CONCEPT



- Sub-system, by Type
- Component/Performance table

Within the SSE, an object is the lowest level construct that is discretely modeled in a given simulation application. Therefore, the object is the lowest level that the user may define employment or deployment specifications. From scenario to scenario, the definition of objects may vary, although the same data definition for an object and the same operational software may be used in different applications.

The most significant departure in the ADISC^2 design from the standard object-oriented approach is in the separation of the functions of an object from its descriptive attributes. Unlike the standard object-oriented taxonomy, the functions of the object in the ADISC^2 are maintained separately with respect to object definitions and are stored as subroutines in a library. The design of the ADISC^2 SSE partitions object movement and operational, or mission, functions in the functional library. This library also is a repository of other activity-related subroutines and algorithms such as processes and utilities.

All motion options are invoked by a single controller (Executive) entry point in the SSE, the Objects Module. It is responsible for moving objects and knowing the location and state values of all structural frames, sub-systems and components operating in a given scenario execution. This promotes efficiency in the re-use of standard routines.

Table structures are also used to support the functional algorithms and provide input values or to serve as a repository for output values during execution. Dynamic subroutines are associated with an object and any supporting data tables through the use of many different types of pointers created interactively by a user when an object sub-system or component is defined that performs a specific dynamic/kinetic function. The construction of these pointers is invisible to the ADISC^2 user.

All performance attributes for objects have at least one initial/default value. Functions are invoked by reference pointers to algorithms and related data tables via a linked list of sub-systems and components interactively constructed by the user as an object type or class is defined.

Using these features of ADISC^2, it is possible to rapidly configure and employ a wide range of simulated objects, assign dynamic representation of varying degrees of fidelity to these objects and move them along pre-scripted routes and perform various actions and interactions with other objects.

The result of this effort has been the design, development and demonstration of a powerful tool with the versatility and flexibility necessary to fulfill a wide variety of simulation and modeling needs. The design allows for varying levels of fidelity depending on the needs of the end user. This design effort has produced evidence as to the feasibility, efficiency and cost-effectiveness in the use of COTS and OTS in a distributed simulation environment. The ADISC^2 effort utilized more than 380,000 lines of legacy software in four different languages (FORTRAN, C, and PASCAL). This capability to integrate existing models into a complex simulation allows the user, for the first time, the ability to incorporate previous efforts without the expense and time associated with transforming the existing databases and algorithms.

Coupled with these design features are a series of powerful applications and control functions that have been developed especially for the ADISC^2 user. These include:

Red Mission Planner. This gives the user a full complement of software options that provide support in performing the tasks of force structuring, deployment, target selection, defense assessment, load out, route planning, weapon/target pairing and attack timing and synchronization. These applications are designed to allow maximum efficiency of user input through easy to follow menu instructions and interactive graphic prompts.

Blue Mission Planner. Here the user is provided with the means to set up a defense scenario and a defensive order of battle. Force structuring, initial deployment, sensor coverage evaluation, attack assessment, redeployment, assignment and reconstitution are accomplished through the menu-driven inputs of the user.

Event-Driven or Time-Stepped Execution. The SSE Executive acommodates either time-stepped, event-driven or hybrid simulation application executions. The current ADISC^2 applications are hybrid only. The simulation controls provided by the Executive allow the user to fast-forward (advance the simulation to a designated point in time and run to that point as fast as the host processor will allow without graphic/animation output), freeze, reverse (either return to a pre-set checkpoint/snapshot or return to a prior event/time-step by re-reading the initialization file and running the simulation in a fast-forward mode to the specified point) and restart (without recompiling) the simulation from any event or point in time within the simulation (requires a hot-start/checkpoint file). Events can be added or deleted and the scenario may be replayed immediately to allow the user to evaluate different defense postures or attack scenarios. Events can be isolated so that direct cause and effect of commanders' decisions can be measured.

Interactive Simulation Execution. Man-in-the-loop decision-making is stressed for many ADI Command and Control concepts. This effort provides the baseline for determining the degree of automation necessary to support the decision-making process. Command and Control of Air Defense requires coordination, direction and control of the activities of many different Air Defense elements, consistent with the time constraints imposed by the threat and necessary response times.

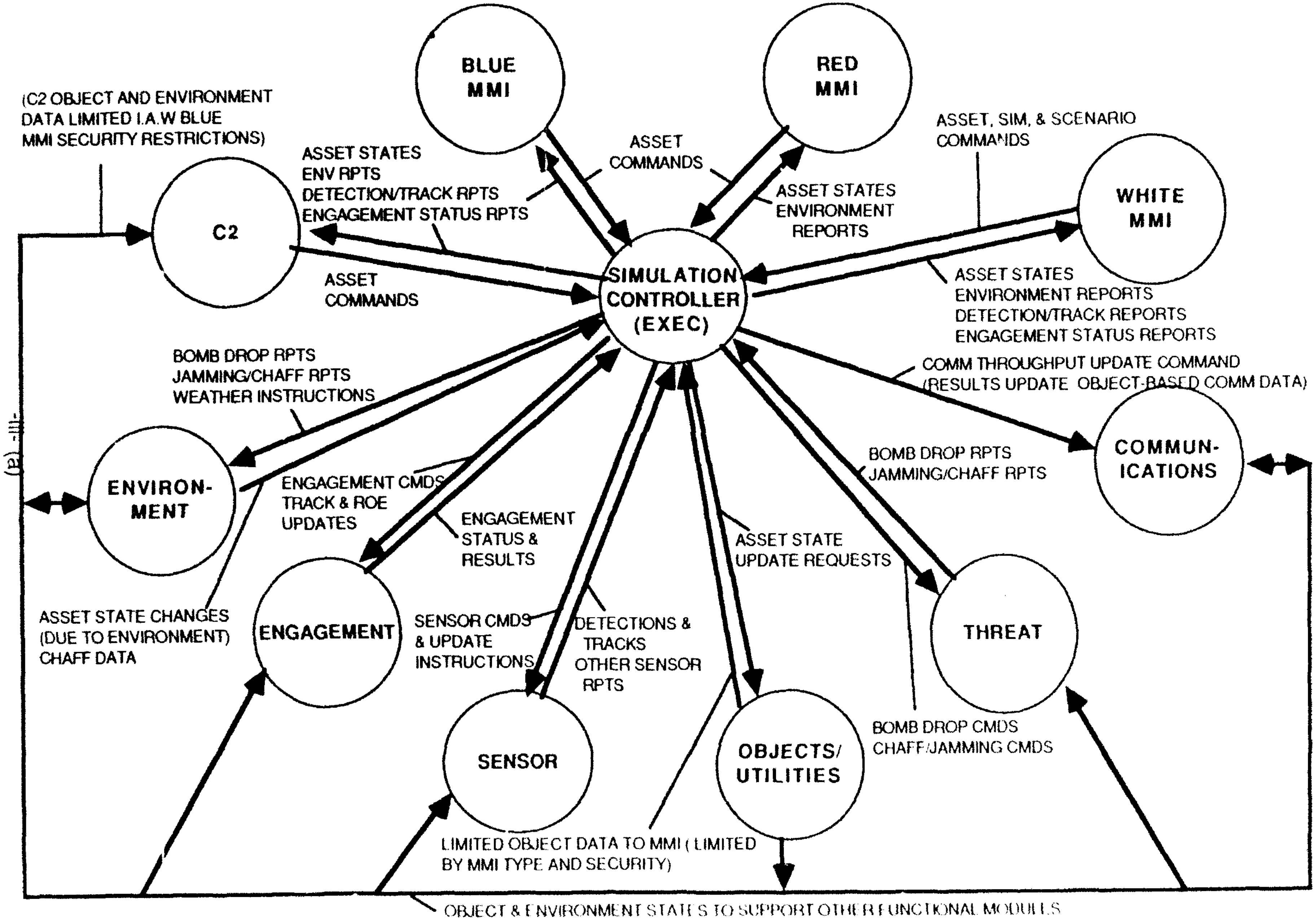
The technology to demonstrate future C^2 functions can be developed and implemented using ADISC^2 simulation, thereby insuring interoperability within a consistent Air Defense simulation environment. These C^2 functions will provide support to the man-in-the-loop as he executes the functions of Air Defense Command and Control. This, in turn, provides a unique opportunity to concurrently develop C^2 functions with the development of advanced technology Air Defense systems.

The basic purpose of the original ADISC^2 contract was to develop a source/sink for simulation of threats, sensors and communications inputs to a SOCC workstation operator in the context of peacetime operations and transition to an enemy attack on North America. The simulation is supported by a flexible, high resolution graphics package. Interactive graphics and display interfaces were prototyped as a current operations baseline to create a realistic representation of air situation data in a manner familiar to a SOCC operator. Through these interfaces, an ADISC^2 operator is able to control all simulated objects. The White Node/Umpire user can also create or delete objects, insert events, inject messages and interrupt the simulation at any point. Figure 2 (on the following page) describes the principle interfaces for the various ADISC^2 functions during run-time.

The C^2 operator, in the simulation role of Surveillance Officer, Identification Officer or Weapons Control Officer at a SOCC can receive information from either simulated or real sensors, communications and intelligence sources and may respond to actions of the simulated threat to the North American continent. The current release of ADISC^2 allows Red players to have access to the "perfect truth" view of the air situation that is available to the White node.

The ADISC^2 also provides the means for interfacing to, and interacting with, other facilities and systems in order to demonstrate, for example, the potential of distributed C^2 operations by placing the operational user in the developmental/evaluation loop.

FIGURE 2 - PRIMARY ADISC2 RUNTIME INTERFACES


BLUE MMI
RED MMI
WHITE MMI
C2
SIMULATION CONTROLLER (EXEC)
ENVIRON-MENT
ENGAGEMENT
SENSOR
OBJECTS/ UTILITIES
THREAT
COMMUN-ICATIONS
(C2 OBJECT AND ENVIRONMENT DATA LIMITED I.A.W BLUE MMI SECURITY RESTRICTIONS)
ASSET STATES ENV RPTS DETECTION/TRACK RPTS ENGAGEMENT STATUS RPTS
ASSET COMMANDS
ASSET COMMANDS
ASSET STATES ENVIRONMENT REPORTS
ASSET, SIM, & SCENARIO COMMANDS
ASSET STATES ENVIRONMENT REPORTS DETECTION/TRACK REPORTS ENGAGEMENT STATUS REPORTS
COMM THROUGHPUT UPDATE COMMAND (RESULTS UPDATE OBJECT-BASED COMM DATA)
BOMB DROP RPTS JAMMING/CHAFF RPTS WEATHER INSTRUCTIONS
ENGAGEMENT CMDS TRACK & ROE UPDATES
ENGAGEMENT STATUS & RESULTS
ASSET STATE CHANGES (DUE TO ENVIRONMENT) CHAFF DATA
SENSOR CMDS & UPDATE INSTRUCTIONS
DETECTIONS & TRACKS OTHER SENSOR RPTS
ASSET STATE UPDATE REQUESTS
BOMB DROP RPTS JAMMING/CHAFF RPTS
BOMB DROP CMDS CHAFF/JAMMING CMDS
LIMITED OBJECT DATA TO MMI (LIMITED BY MMI TYPE AND SECURITY)
OBJECT & ENVIRONMENT STATES TO SUPPORT OTHER FUNCTIONAL MODULES

The man-in-the-loop evaluation process provides the capability for evaluation and analysis of C^2 technology for the ADI as well as providing a vehicle for operational-user involvement within the C^2 development process. In addition to Man-in-Loop, the simulation can accommodate software and hardware in the loop operations.

Ability To Rapidly Integrate Legacy Software As Reuseable Scenario Elements. The Government has spent millions of dollars in the development of simulation software. Much of this software is characterized as machine dependent, special purpose and limited in scope, granularity and fidelity. Some models are quite useful but have extremely user-unfriendly interfaces. Many of these legacy software products have achieved a high level of user confidence over years of successful application. The ADISC^2 Simulation Support Environment makes possible the cost-effective integration of legacy software.

The current design reflects a commitment to Open-System architecture which reduces the machine dependency of the software legacy and makes its integration and re-use a timely, cost-effective alternative. Existing models are surveyed to find algorithms and subroutines that represent fundamental laws of physics that govern functions. operations and processes performed on and by objects. These software elements are then archived in the Simulation Support Environment as part of the library of available object dynamics, mission and operational functions and utilities.

By implementing simulation execution as a message-passing exercise, controlled by the Simulation Executive, it is possible to define messages to invoke calls to these subroutines, to pass parametric data, and to drive tabular and graphic displays as the simulation is executing. This allows great flexibility in defining objects and control interfaces to perform a broad range of varied functions, within both current and future asset performance envelopes, at varying levels of simulation fidelity in each scenario.

Automated Command & Control Options. For many analytical and evaluation activities supported by the ADISC^2 Simulation Support Environment, the interactive participation by human subjects is undesireable. Martin Marietta was commissioned to develop alternatives to man-in-loop simulation executions. This enhancement was accomplished while retaining or expanding the basic sensor, threat, communications, environment, intelligence and air traffic capabilities of the original ADISC^2.

Modifications were made to the ADISC^2 that allow users to create, archive and modify Rules of Operation and Rules of Engagement for use in ADI scenarios. The design is based upon a stimulus-response model of controller activities. The SSE provides an interactive interface to input definitions and specifying thresholds for stimuli and appropriate responses. An example of a user-selected response might be: "Activate a CAP with a (user defined) mix of surveillance, engagement, and support (e.g. interceptors and tankers) elements at a (user defined) location and (user defined) patrol pattern".

The initial baseline set of air defense rules was developed by the Martin Marietta program staff with the participation of Air Force representatives experienced in command and control, including officers and non-commissioned officers currently assigned to air defense operations as well as ESD and RADC technical staff. The design focuses upon individual C^2 site activites of surveillance, identification and weapons control and the associated higher-level activities involved with evaluating the threat and implementing operational plans. Figure 3 (page iv(a)) lists the high level functionality of the resulting module. With this capability invoked, it is possible to pre-script all control and directing activities for all defense elements and to execute end-to-end engagement scenarios, with or without Red interactive control of threat forces, and with or without exercise of Blue operator over-ride options at any point in the execution. It is possible for Blue operators and the automated C^2 module to

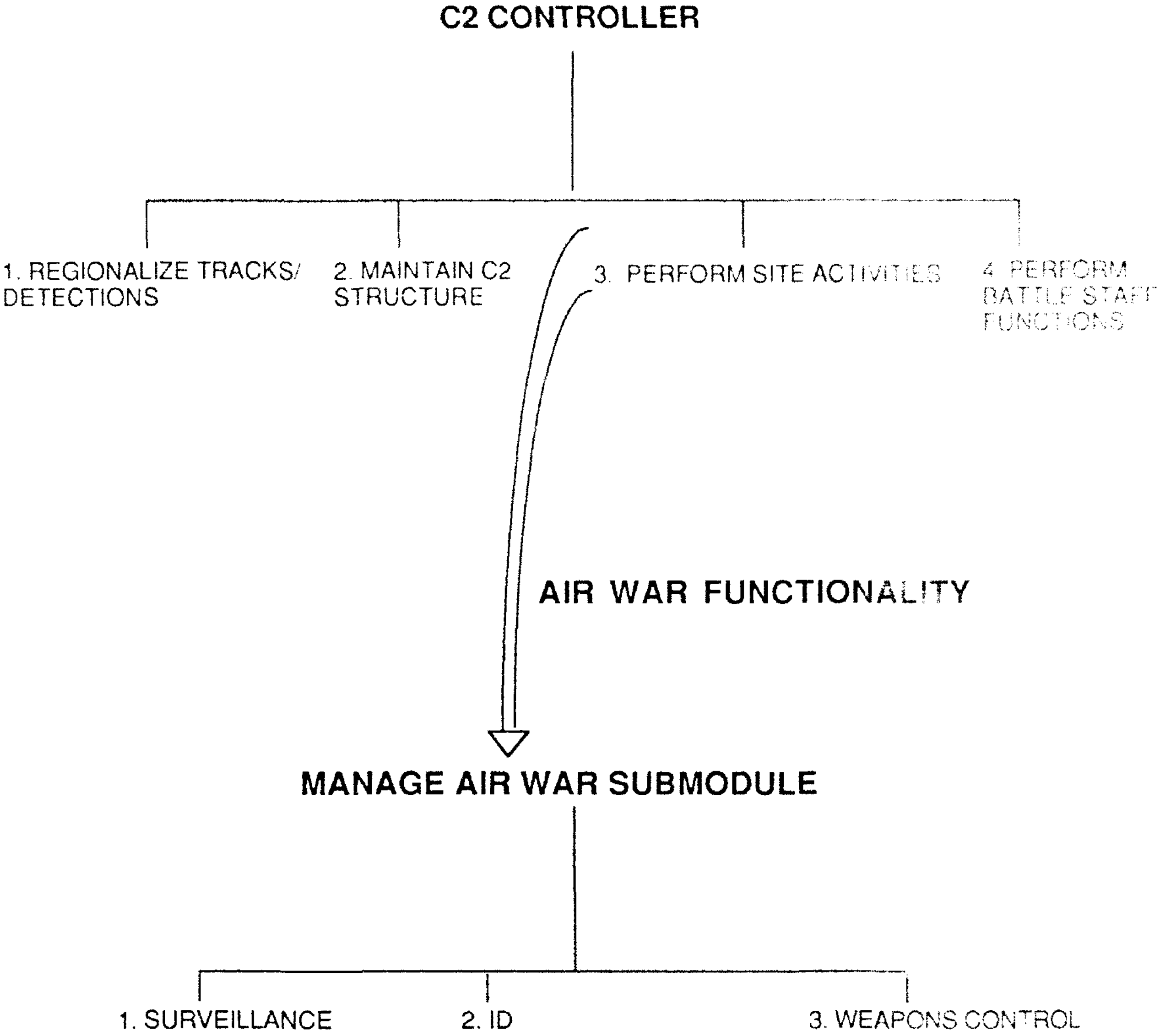


FIGURE 3 - C2 FUNCTIONAL REPRESENTATION

participate in the same scenario execution.

Analytical Module. Under the same change order to the base ADISC^2 contract that directed the development of the Automated C^2 Module, an Analytical Module was added to allow analysts to rapidly and efficiently exploit data derived from ADI simulation executions. The primary purpose for the Analytical Module is to allow the analyst to compare performance of alternative defense architectures and operational C^3 ADI defense concepts against various threat scenarios. A robust package consisting of most standard statistical applications (eg., linear regression, multiple regression, sorting utilities, T-tests, Chi-squared analysis, etc.) was integrated into the SSE using a COTS product provided by the Numerical Analysis Group, of Oxford.

Additional modifications were made to the Pre-simulation Module of the SSE that allow the user to create, archive and modify data collection specifications to be used in scenario executions. Users may identify data of interest from the standard messages used in the simulation environment. The user may specify the collection interval or frequency for any data item and may invoke collection specifications for any object, category, type, class, system or sub-system. This allows the user to achieve desired levels of data collection and storage during the simulation and minimizes impact on execution speed. Figure 9 (see page 9c of this report) illustrates the post-processing activities that are used to support the user in exploiting data collected during simulation execution.

Overall Performance. With all of its capability, ADISC^2 is able to perform complex simulations, involving relatively large numbers of objects, at real time or at speeds greatly exceeding real-time. It is possible to host ADISC^2 software on a single graphic workstation. The preferred three-node hardware architecture hosts the ADI simulation and the INGRES relational database on one SUN 4/280 workstation, with a White/Red Man-Machine Interface (MMI) hosted on a second SUN 4/260 node, and a Blue MMI resident on a third SUN 4/260 workstation. These three workstations are connected via a local area network operating under standard ethernet protocols. The Pre-simulation Module offers the capability to reconfigure the workstations on the network to be either Red, White or Blue; to perform specific simulation functions or roles (eg., a node can be designated to perform the functions or support the display of data from a single sensor, all sensors or a specific SOCC Surveillance Officer's station); or to support geographically distributed simulations across local or wide-area networks.

Presently, the ADI scenarios performed by the ADISC^2 include a threat of more than 1800 air-launched and submarine-launched cruise missiles and hundreds of enemy aircraft flying from potential enemy attack bases located in various parts of the world. These simulations include over 200 radar sites and more than 2500 potential targets. Each radar site performs surveillance functions at its normal scan rate and the computer performs simulation of these radars at the signal-to-noise ratio level, responding to changes in altitude and radar cross-section (RCS) of each object within its field-of-view. These scenarios also include sensor and communications jamming and deception tactics, simulation of link-level propogation of communications, with nuclear and environmental effects. Scenarios can be constructed for joint service operations involving Air Force, Navy and Army elements.

In the current hardware/software environment, with limited attack scenarios and with perfect communication, it is possible to achieve execution speeds of 200 to 800 times faster than real time. With nuclear effects, chaff clouds, and sensor jamming, it is still possible to achieve executions about two to three times faster than real time on the ten-million-instructions-per-second (MIPS) workstations of the current architecture.

INTRODUCTION

This document summarizes the work accomplished under Contract F30602-87-C-0221, Air Defense Initiative Simulation for Command and Control (ADISC^2) Development, covering the Phase I period of performance from September, 1987, to June, 1988, and the Phase II period of performance from June, 1988, to October, 1990. The original end of contract was December, 1989. However, as a result of several Engineering Changes to the base contract, both the tasking and period of performance of the origional contract were modified.

The objective of this contracted effort was to design, implement, install, test, operate and maintain an integrated simulation to provide Threat, Surveillance, Engagement, and Communications models which will be exercised in accordance with Government-provided threat scenarios.

The long-term objective of the ADISC^2 is to provide a consistent baseline to develop the technology to demonstrate man-in-the-loop C^2 functions in support of the Air Defense Initiative (ADI).

Martin Marietta Information & Communications Systems, under contract to RADC/COAA has developed a simulation capability providing a consistent baseline for development and evaluation of potential applications of command and control (C^2) technology for strategic air defense requirements. This development incorporates functional modules of **Threats** (bombers, air-launched cruise missiles, gravity bombs, submarines, sub-launched cruise missiles, electronic jammers and tankers), **Sensors** (ie., JSS, NWS, Alaskan radar, OTH-B, SEEK SKYHOOK, Space-Based Radar, SOSUS, SURTASS, and airborne and shipborne radars, to name a few), **Engagement** elements (F-15 SAMs, etc.), and representation of **Environment,** and **Communications**. An **Air Traffic** Module is provided to allow realistic simulation of background activities (commercial, military airlift, & Strategic Air Command) in support of air defense engagement scenarios.

Based on convictions derived from years of prior simulation experience, the Martin Marietta ADISC^2 design solution recognizes that modern C^2 simulations should have the following characteristics:

1) Portability of software within an "open-system" architecture; i.e., independence with respect to discrete machines or languages;

2) Moduiarity and flexibility achieved within a total "Simulation-Support Environment", where functions (new and legacy) may be incorporated;

3) Capability to insert application modules of varying fidelity without major recode;

4) Expandability in the number of application software modules that can be installed and remain transparent to the user;

[1] Rome Air Development Center, 'Statement of Work for Air Defense Initiative Simulation for C2 Development' (ADISC2), PR No B-7-3134, 14 May 1987

5) Ability of the SSE to support interrupt-driven, event-driven, time-stepped or hybrid execution of scenarios (current ADISC^2 applications are hybrid only);

6) Ability to be interrupted, restarted, or backed up and inject changes in object status, in dynamic interaction or in simulated events without re-compiling or re-linking;

7) Capability for man-in-the-loop interaction, including dynamic object control and injection of objects and events.

OVERVIEW OF THE ADISC^2 DESIGN

In order to meet the rigorous requirements set forth for the ADISC^2, Martin Marietta's design approach rejected the traditional simulation solution of hard-coding an ADISC^2 Engagement Model in favor of providing a more modular and flexible Simulation Support Environment (SSE) that is populated with ADI-specific applications. The SSE provides services to the user during Pre-Simulation, Runtime and Post-processing Analysis modes of operation.

Special design features which greatly enhance the capabilities of the ADISC^2 testbed include:

- Distributed Simulation–geographically and functionally;

- "Open" Architecture;

- "Plug-In" Modularity of Functions & Re-use of Legacy Software Without Transliteration of Code;

- Maximum Use of Commercial-Off-The-Shelf (COTS) and Off-The-Shelf (OTS) Software;

- Flexible Configuration Control;

- Compliance with Current and Emerging Software Standards

Operating System:	UNIX/POSIX; IEEE Portable O.S. Std.
Relational Database Management:	SQL; ANSI Standard
User Interface Management System:	BLOX; OSF Motif Industry Std.
Window Management:	MIT, X Consortium
2D and 3D Graphics and Animation:	FIGARO/PHIGS; ANSI Standard
	FIGARO/PEX; PHIGS Ext. to X-Window
Networking:	ETHERNET; IEEE Low-level LAN Std.
	TCP/IP; Industry Std. Comm. Protocol
Programming Languages	FORTRAN, C; ANSI Std.

- Guaranteed Interoperability with the Strategic Defense Initiative (SDI) National Test Bed (NTB);

- Object-Oriented/Artificial Intelligence Compatibility; and

- Rapid Prototyping.

The design is based on message-passing and shared-memory paradigms. Messages used in the ADISC^2 design are true in format, content and data rate to messages used in real-world command and control, sensor and communications systems. This allows relative ease in configuration of distrubuted hardware/software architectures with real and simulated nodes participating in the same executions.

Within the ADISC^2 SSE, an object is defined in a manner consistent with the Entity Attributes-Relational Attributes (EARA) model employed in most standard Object-Oriented Analysis applications. This allows logical structuring of the attributes of objects into sets unique to specific categories, classes, systems, sub-systems and components. Under this design, the table-building, archive management, and data access tools provided by the COTS relational database management software, are used in a pre-simulation interface mode to allow the user to create, employ and maintain tables of object attributes. The use of utilities provided by a relational database management system (DBMS) is significant in that the table-building utility offers virtually unlimited expansion of the breadth and depth of attribute descriptors for any object with a minimum of administrative overhead and database maintenance.

Tables may be created to incorporate and structure sets of attributes that describe objects. Tables incorporate attributes that uniquely identify objects, establish performance envelopes for objects, define the operational or mission environment of objects, or establish the basis for potential action/interaction among objects, systems, sub-systems, or components.

The logical layering used in the ADISC^2 design for object definition is:

- Structural Frames within an environmental medium
- Category
- Class
- Object
- Sub-system, by Type
- Component/Performance table

Within the SSE, an object is the lowest level construct that is discretely modeled in a given simulation application. The object is the lowest level that the user may define employment or deployment specifications. From scenario to scenario, the definition of objects may vary, although the same data definition for an object and the same operational software may be used in different applications.

Objects are defined first as structural frames operating in one of several environmental media (spaceframes, airframes (aircraft or missiles), ground structures (fixed or mobile), ships/boats (surface or sub-surface vessels). In constructing a class of objects, the user must input those parametric values corresponding to a specific set of attributes common to all such structural frames. These attributes are defined as those necessary for the object to conform to the pertinent, fundamental laws of physics that apply to all bodies that exist or move in the environmental medium. Classes and objects within classes inherit the data formats for the set of all attributes defined for their environmental medium. Within a class, all objects will have the same parametric attribute values and carry the same sub-systems and components.

A "class", therefore, is a collection of objects that share the same set of descriptive attributes and that can perform the same set of functions. Performance attributes are only defined at the class or platform level and the initial attribute values are inherited by all sub-classes and objects/assets within a designated class.

Super-sets of classes may be created, called "categories". Categories include all objects within various classes having the same sub-sets of attributes, irrespective of the parametric values that define the different objects comprising each class. An example would be a category of "interceptor" that includes classes of F-15, F-16, F-14, F/A-18, etc. In the ADISC^2 design, the standard class definition has also been refined by introducing the concept of "structural frames", "systems/objects", "sub-systems" and "components" to allow the object-oriented taxonomy to adapt more easily to the strategic and tactical environment that the SSE currently serves.

In most standard object-oriented applications, the functions of each object (eg., VERB-like elements) are treated as attributes of the object and are, thereby, "hard-coded" as part of the object definition. This feature of standard object-oriented design was determined to be too restrictive to accommodate the requirements for flexibility in incorporating current and future performance envelopes of objects and it did not allow varying levels of fidelity to be represented in simulating object action and interaction for ADI evaluation.

The ADISC^2 Simulation Support Environment (SSE). The ADISC^2 SSE design has, as its centerpiece, the Strategic Defense Initiative (SDI) National Test Bed (NTB) Simulation Executive, thereby assuring interoperability with SDI applications.

To provide a comprehensive, user-friendly simulation environment built around the functions of the Executive, requirements were identified for additional simulation support functions such as definition of simulation and run-control parameters, selection of simulation objects, creation of simulation scenarios, display generation, and others. Related functions were organized into modules that interface with and support the ADISC^2 user. These modules are presented in Figure 4 as an ADISC^2 SSE functional diagram including the following:

Pre-Simulation Module:

- Administration Functions (including external system interfaces)
- Prototyping Options
- Scenario Generation Functions
- Run Configuration

Relational Database Management System

- Scenario Archive Maintenance
- Object Definition Functions
- Cartographic Data Management
- Functional/Utility Library Archive Management

Man-Machine Interface

- Pre-simulation Interfaces to the Relational Database Management System
- Rapid Prototyping Interfaces
- Interactive Display/Control Interfaces
- Cartographic Management System
- Graphics/Animation Management

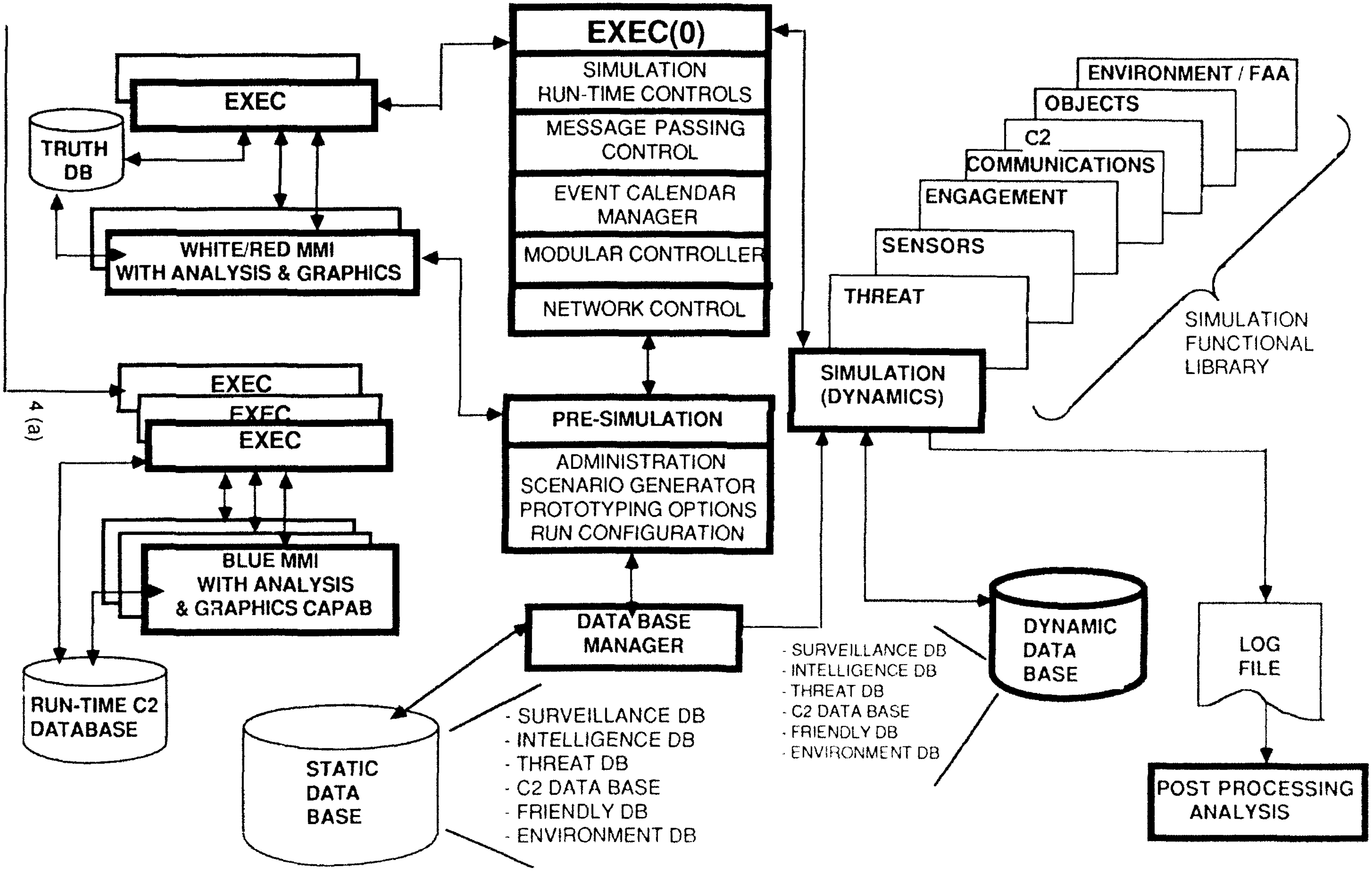
Functional Modules:

- Threat
- Sensors
- Engagement
- Communications
- Environment
- Objects/Utilities
- Automated Command & Control

Runtime Controls/Simulation Executive

- Interfaces to the Network Database Management System
- Simulation controls
- Timing and Synchronization
- Network Management Interfaces

FIGURE 4. - ADISC2 DYNAMIC SIMULATION FUNCTIONAL DESIGN CONCEPT



- Operating system Interfaces
- Message Management

Post-Simulation Analysis Module

The SSE is comprised of several integrated Commercial-Off-The-Shelf (COTS) products such as INGRES for relational database management, FIGARO for animation and graphics, X-Windows and Sun-tools for window management, Template/Blox for user interface management, and NAG for statistical analysis. This powerful composite provides a highly flexible environment that allows the user to rapidly configure simulations to a wide range of operational evaluation and analytical requirements.

The ADISC^2 Simulation Executive. The ADISC2 Simulation Support Environment places all of the simulation controls, application module controls, timing and synchronization, and other functions common to all simulations under the control of an ADISC^2 Simulation Executive. This Executive was originally developed for the SDI National Test Bed program. It is machine independent and provides the user with the control functions of:

- Initialize / Restart
- Pause
- Resume
- Checkpoint / Backup
- Programmable Snapshots
- Event / Message Injection
- Clock Control
- Fast Forward/Reverse
- Functional Module Control
- Scheduling & Execution Management
- Archive and Replay
- *Message Passing*
- Operating System & Network Coordination
- Stop.

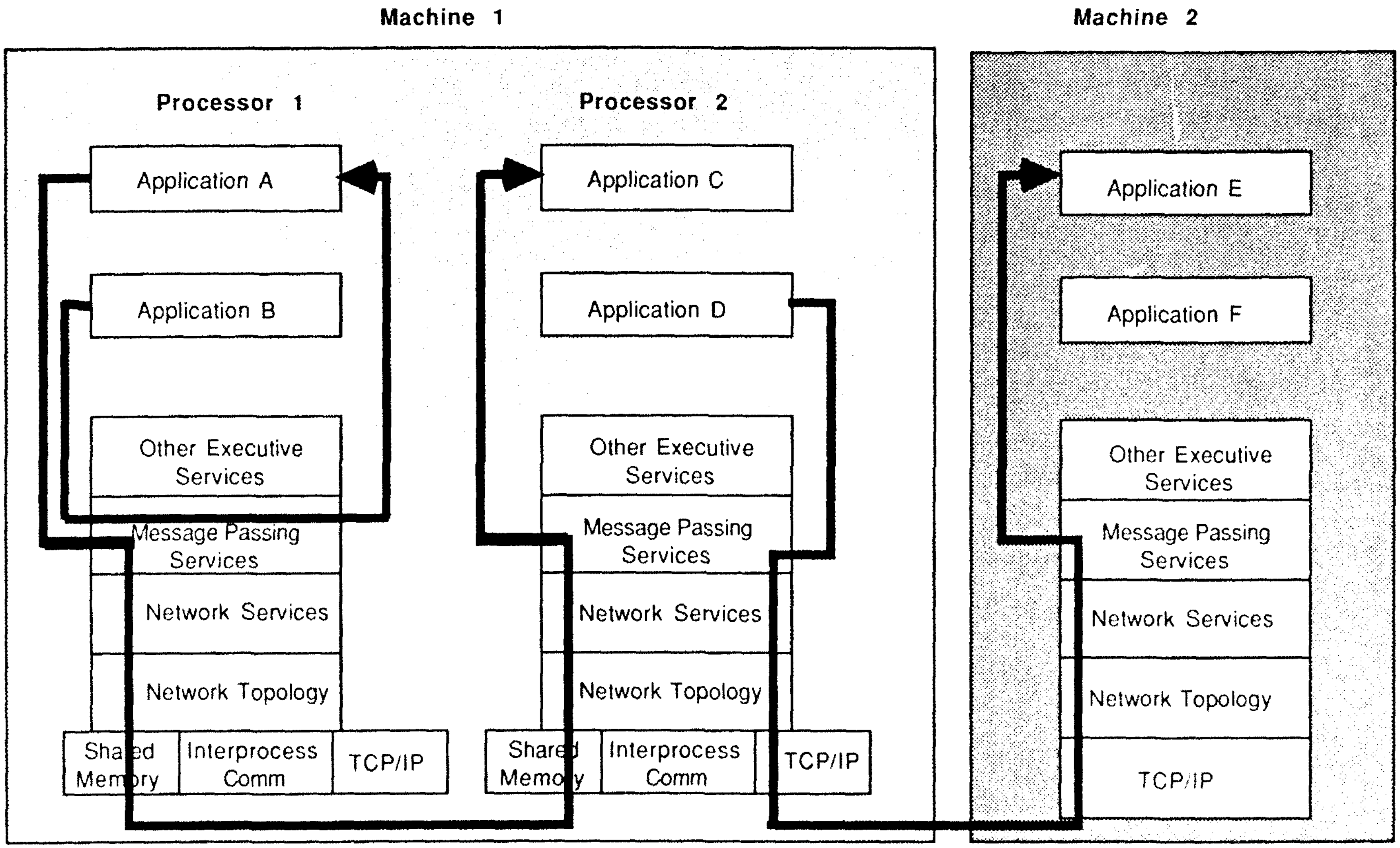
This Executive provides the ADISC^2 with state-of-the-art distributed, parallel processing simulation capabilities. Under this Executive, all simulations are performed according to a simple message-passing paradigm, using an interface control structure to specify message format and content (See Figure 5 on the following page). Where there is a correspondence with messages used in the real world, messages used in the ADISC^2 Simulation Support Environment (SSE) are constructed in a manner to insure that identical content, format and rate of presentation are maintained.

The Simulation Executive provides a flexible, transportable, distributed simulation controller for the ADISC^2. This Executive controls the scheduling of, and coordinates the inter-process communication between, simulation subroutines (simulation models, functional/dynamic applications, and/or software utilities) within the SSE. These simulation routines may be spread across multiple computers and across multiple central processing units (CPUs) on computer platforms with multiple processing capability.

Simulation subroutines communicate within the SSE by passing messages using services supplied by the Executive. The Executive coordinates message transfer between subroutines, including transfers between software elements that require message transfers between computers/CPUs. The functional modules and simulation subroutines that are communicating between CPUs via message-passing never need to know on which computer, or under which Executive, any other subroutine is operating so long as the common-block definitions are resident on both machines.

The Executive controls the scheduling of processes and, therefore, is required to

FIGURE 5. SIMULATION CONTROLLER PROCESSING



APPLICATION MESSAGES ARE PROCESSED AND TRANSMITTED VIA STANDARD METHODS USING SHARED MEMORY OR TCP/IP maintain system clocks. It maintains global synchronization of the pseudo-real-time clocks across all computers participating in the simulation environment. This allows the simulation time-frames to advance concurrently, thereby coordinating the scheduling of application routines running on different computers.

The Executive optionally maintains performance statistics and provides on-request data dumps to facilitate debugging and to assist in optimization of the simulation environment and its application programs. Application programs may enable or disable collection of execution-time and message-traffic statistics. Data dumps may be requested selectively; periodically, at the start, and/or at the end of a simulation run.

To support these functions, the Executive must be provided with certain characteristics of each of the subroutines that it will be controlling. The Man-Machine Interface provides a flexible interface between the user and configuration management features of the Executive for definition of these characteristics. The values obtained through this interface are passed through data files to the Run-time Controller portion of the Executive for use during simulation execution.

The Executive coordinates network and operating system interfaces, manages schedules and maintains event calendars, maintains timing and synchronization of nodes within the network, performs logging functions, and provides run-time interactive controls. The ADISC^2 design allows interactive, interruptable, real-time (variable time-ratio control), event-driven, interrupt-driven, or time-stepped simulation execution supporting man-in-the-loop, hardware-in-the-loop, or software-in-the-loop.

ADISC^2 Man-machine Interface. ADISC^2 also incorporates COTS graphics software to provide Rapid Prototyping of multi-color displays with full 2-D and 3-D animation. A Man-Machine Interface (MMI) provides user-friendly access to the Pre-Processing, Run-time and Post-Processing controls of the ADISC^2 including Red Mission Planning (Force Structuring, Deployment, Target Selection & Prioritization, Route Planning, Weapon Load-out, Weapon/Target Pairing, Defense Assessment, and Attack Timing and Synchronization), Blue Defense Planning, pre-scripted and dynamic event injection, and dynamic object creation/deletion.

The rapid-prototyping capability included in the design allows the user to define new objects or to create and employ alternative dynamic/functional subroutines of varying fidelity. Using this rapid prototyping capability, users can build operationally familiar interactive control displays to move simulated objects and/or cause them to perform functions such as sensing, interception, or other interactions. Prototypes can also be constructed to define tabular or graphic displays to be used and posted dynamically during the simulation.

The ADISC^2 Statement Of Work (SOW) required the provision of real-time, interactive graphics using Commercial-Off-the-Shelf (COTS) software. The proper selection and employment of COTS graphics and animation software allows a hardware-independent design solution that complies with accepted industry and committee-adopted standards. A trade study was performed early in the period of performance to evaluate commercially available candidate graphics and user interface management systems (UIMS) products. The results of this trade study led to the selection of two separate software packages offered by Template Graphics Software, Inc., of San Diego. TEMPLATE/BLOX was selected for the UIMS and FIGARO for real-time graphics and animation.

Separately, these two products were roughly comparable to most of the other vendor offerings according to the criteria established to evaluate and select graphics and UIMS software. At the time of the study, no COTS product could provide rapid-prototyping and meet the requirements for real-time update of graphic displays. Further, no COTS product could be found that supported multiple processes required for graphical updates by the simulation while simultaneously accepting interactive user input.

However, the combined capabilities of the two products offered significant advantages and features not available from any of the commercially available contenders. Template Graphics Software, Inc. had tentatively planned to integrate the features of these separate products over a period of two years. Martin Marietta's program staff working closely with TGS technical staff, was able to accomplish the planned integration of these two packages in a period of about two-and-one-half months.

ADISC^2 Database Design. The ADISC^2 employs relational databases to maintain scenario generation and configuration data during Pre-simulation. A relational database is perceived as a collection of tables. The tables in the database contain a specific number of columns with a specific format. But the number of rows of data in the table is not limited. A user staff member designated as the Database Administer may add tables and columns easily.

A COTS product, INGRES, was selected to Support the Pre-simulation functions of the ADISC^2 SSE. It provides two query languages, Query Language (QUEL) and Structured Query Language (SQL). The ADISC^2 applications are implemented in QUEL. However, either language may be used to access the ADISC^2 databases. This provides great flexibility in allowing the user to rapidly integrate data from other commercial relational database management products such as ORACLE and SY-BASE by accessing the standard relational constructs at the SQL level.

The ability provided by INGRES to create interfaces to its relational database management utilities using most high-level software languages (including FORTRAN and C) insures that familiar, user-friendly, operational interactive interfaces to the data can be developed under the rapid-prototyping features of the MMI. The ADISC^2 Pre-simulation programs provide a forms-based interface to these databases which are accessed for updates and for initialization of simulation parameters. The databases may be accessed through any of the INGRES front-ends, as well as the ADISC^2 application programs.

The design of ADISC^2 incorporates a Library Database. This database maintains the configuration and administrative data for ADISC^2. The design also includes a Scenario Database. This database maintains the scenario generation and object prototyping data for each simulation.

The ADISC^2 design also employs a network database management system (NDBMS) for support of message passing activities under the control of the Simulation Executive during runtime. Each CPU in the ADISC^2 architecture has a healthy allocation of internal memory (minimum 32 mega-bytes each, with 64 mega-bytes preferred for most platforms hosting a full complement of MMI software and 128 mega-bytes recommended for the network server node).

The runtime data management system of the ADISC^2 uses available caches of memory to support two primary runtime-support activities: 1) maintenance and management of message queues for messages transferred between CPUs; and 2) for storage and management of arrays used at the application level, under local process control, to describe objects, system and sub-system instances/states as the simulation is executing. In the second case, exploitation of shared memory is accomplished in a manner much like shared-common buffers are employed by FORTRAN programmers. Each processor constructs messages, transmits and receives messages through services provided by the Executive (including queue management services), receives and processes messages of interest, and uses internal addressable memory to store and maintain arrays of attribute status flags describing, among other things, the instances of objects operating in the simulation.

ADISC2 OPERATIONAL MODES

ADISC^2 Pre-Simulation Functions. The ADISC^2 Pre-simulation function (See Figure 6) allows the user to select the level of functional representation required to achieve the speed of execution and the level of granularity necessary for his analytical needs. The user may mix-and-match various scenario elements such as archived run-control specifications, network configurations (number of nodes may be varied and Red/Blue/White configuration may be varied), functional role of the node may be varied, threats (number & type of threat objects, routes, mission objectives and activities may be varied), defense configuration, and pre-scripted events may be created, modified or invoked. This allows the user to rapidly configure baseline scenarios and examine a broad range of excursions from a given baseline to support a wide variety of simulation activities.

The design of ADISC^2 pre-simulation functions recognizes that the user community is likely to be quite diverse. At the time of this writing, there were three ADISC^2 users; RADC, ESD, and NORAD/J-5 (a total of eleven Government agencies have expressed an interest in migrating the ADISC^2 software to support various applications, including Air Force Air Staff, Army Systems Command, and others). ESD/XRT has assumed responsibility for configuration control and maintenance of ADISC^2. RADC will continue to be the contracting agency. Martin Marietta will be the ADISC^2 engineering and development contractor. Those wishing to use this software or to add new applications will coordinate requests through ESD/XRTI staff.

There will be ESD staff and staff at the various Government sites who will be responsible for the maintenance, configuration management and operational control of ADISC^2. These users are primarily interested in the integrity and modularity of the software, file structures, data access maintaining an efficient baseline hardware/software configuration, maintain g local scenario archives, maintaining proper security configuration (generally, issues related to the sufficiency of the environment to perform simulation efficiently and effectively).

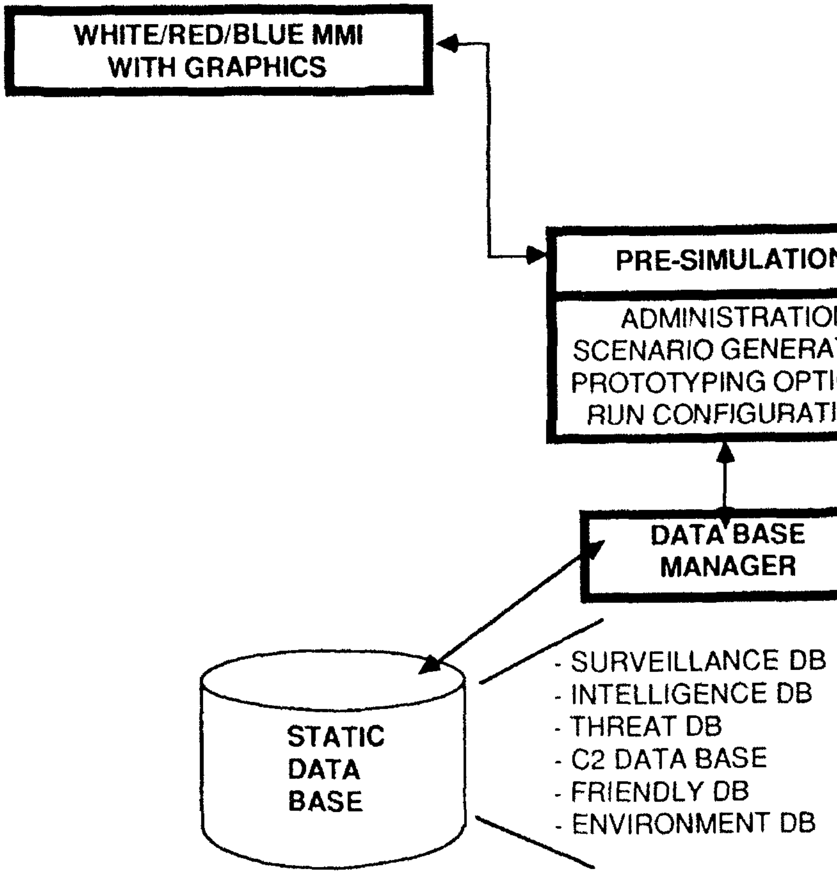
Other users will be responsible for design, development and implementation of evaluations and experiments that will be supported by ADISC^2. These users will be primarily interested in an interface that will be called "user-friendly" if it provides effective and efficient support in generating alternative scenarios, rapidly mixing and matching ADISC^2 elements, archiving and returning previously executed data for modification and/or comparison and analyses.

This view of ADISC^2 "user-friendliness" would have little meaning for a third set of potential users who are primarily interested in adding a new class of weapon or sensor system or sub-system to ADISC^2 in order to determine the impact of new technology on C^2 for Air Defense.

A "User-friendly" interface for this user group would be one that provides rapid and easy access to the object-oriented database, helpful access to the dynamic utility library for inserting software (in a "plug-in" concept that allows new models/functions to be added), and/or the ability to generate new graphics displays, status boards or input tabular output report specifications necessary to the incorporation of a new class of ADI-related elements. Potential users with this type of orientation are primarily focused on the features of ADISC^2 related to flexibility and responsiveness in incorporating new elements of an ADI simulation to support scenario generation. As such they may or may never have need to invoke those functions of ADISC^2 that are related to actual scenario generation or ADISC^2 administration.

The last set of potential users may be operational users, test subjects, or other groups who just want to turn ADISC^2 on and participate passively or interactively in one or more simulation executions. These users would consider the support functions of administration and scenario generation or rapid prototyping as anything but

FIGURE 6 - ADISC2 PRE-SIMULATION FUNCTIONAL DESIGN CONCEPT



- CONFIGURE HDW & SW
- SYSTEM SECURITY
- MAINTAIN ARCHIVES
- SET UP DATABASE
- SET UP SCENARIO

"user-friendly" if they were required to perform any of these functions not required by their task or interest in ADISC^2.

Recognizing that there are many views that constitute "user-friendliness", depending on the type of intended use or interest by the user community, the pre-simulation interface is organized to support different needs of the RADC staff; allowing directed and computer-supported access by a wide variety of potential users.

Runtime Simulation. The runtime mode of ADISC^2 operations is comprised of the functions required for system control by the White, Red, and Blue node operators during the running of the simulation, including the interactive menus and controls. Figure 7 (on the following page) describes the functional relationships among the elements of the SSE that are invoked during the initialization mode between Pre-simulation and simulation execution activities. Figure 8 illustrates the functional relationship during execution of an interactive simulation.

Post-simulation Analysis provides statistical analysis including graphical representation of selected data collected during execution of a scenario or a series of scenarios. In general the sequence of operation would be: Setup the database, scenario, equipment configuration, and software configuration within the pre-simulation portion of ADISC^2 system; run the simulation with the runtime simulation controls; process the data collected using the post simulation analysis functions. Figure 9 presents the sequence of functions involved in the data analysis support provided by the SSE.

While this activity is routinely utilized at the end of a scenario execution (thereby labeled a Post-processing activity), the user may interrupt the execution at any point and perform analysis on data collected to that point. This feature is particularly useful when combined with the control functions provided by the Executive to checkpoint to a prior event or time-step and repeat portions of the scenario execution with controlled changes. With changes to the inputs, the simulation outcome may be perturbed. If the user pauses the simulation at the same point in the second execution, analytical processes may be repeated on the partial scenario data reflecting changes made. By comparing the differences in the two outputs, it may then be possible to evaluate sensitivity of the overall simulation outcome or to examine trends relative to changes made in partial scenario executions.

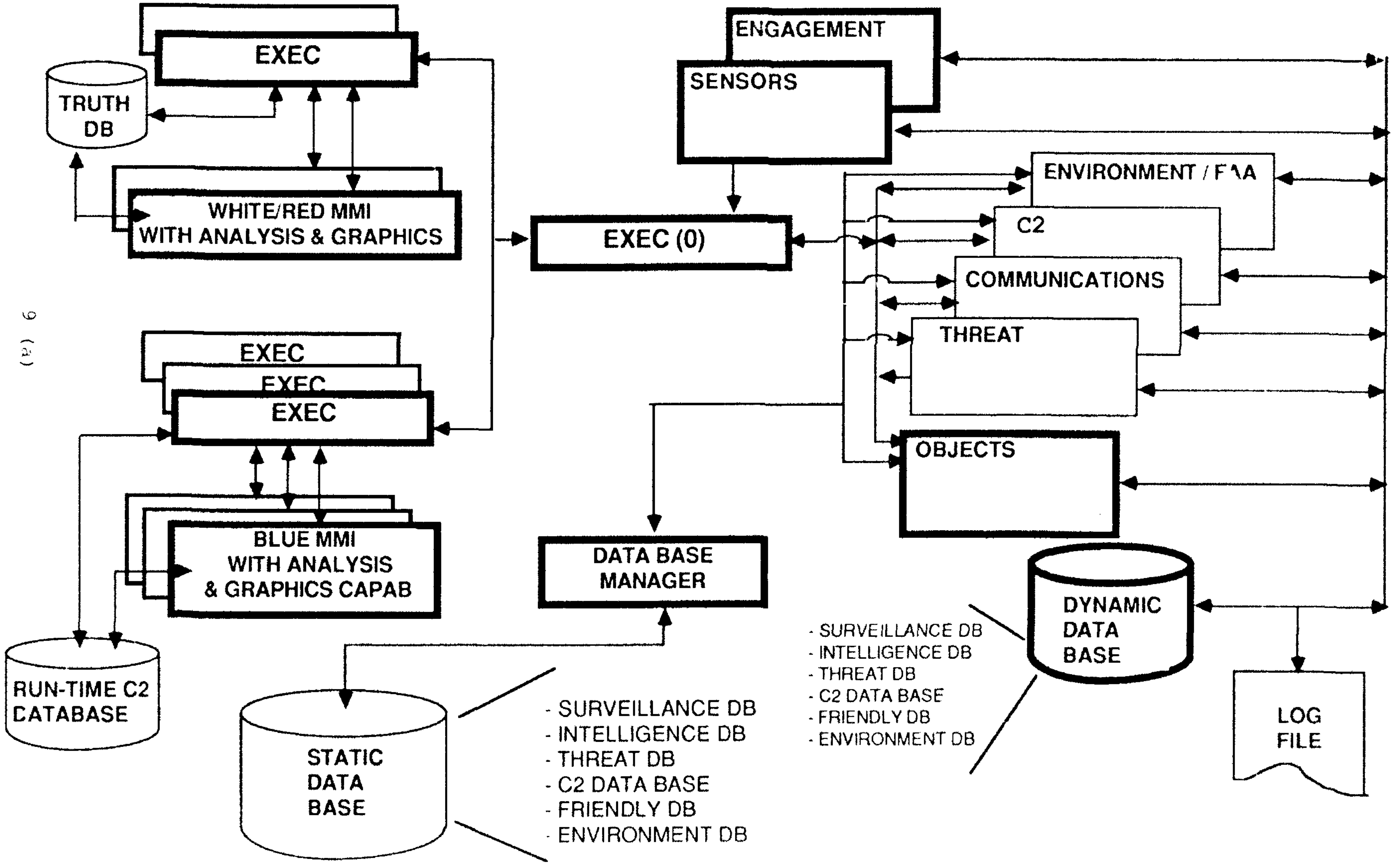
ADISC^2 OBJECT-ORIENTED DESIGN CONCEPT

The flexibility of ADISC^2 is made possible, in part, by the object-oriented concepts employed in the design of the SSE. Object-oriented design allows users to create a wide range of simulated threat and defense elements by mix-and-match of physical and performance attributes. Object-orientation also allows the simulation to be compatible with most Artificial Intelligence applications for command & control. To understand the full potential of this environment, it may be helpful to understand the unique manner in which object-oriented design constructs were employed in ADISC^2 development.

Most object-oriented approaches use a taxonomy that allows fairly obvious analogies to the English language and its grammatical constructs. Objects (NOUN-like elements) are defined as a collection of associated attributes (ADJECTIVE-like data elements) that are used to describe the object and bound its performance.

The "domain" of an object is generally defined as the universe of all related attributes for a given object. The "instance" of an object is the collection of parametric values of all state attributes within the domain of an object at a particular instant in time. In most object-oriented paradigms, definition of objects involves utilization of attributes that allow the unique, mutually exclusive identification of an object.

FIGURE 7. -   ADISC2 SIMULATION INITIALIZATION FUNCTIONAL DESIGN CONCEPT



)

FIGURE 8. - ADISC2 SIMULATION EXECUTION FUNCTIONAL DESIGN

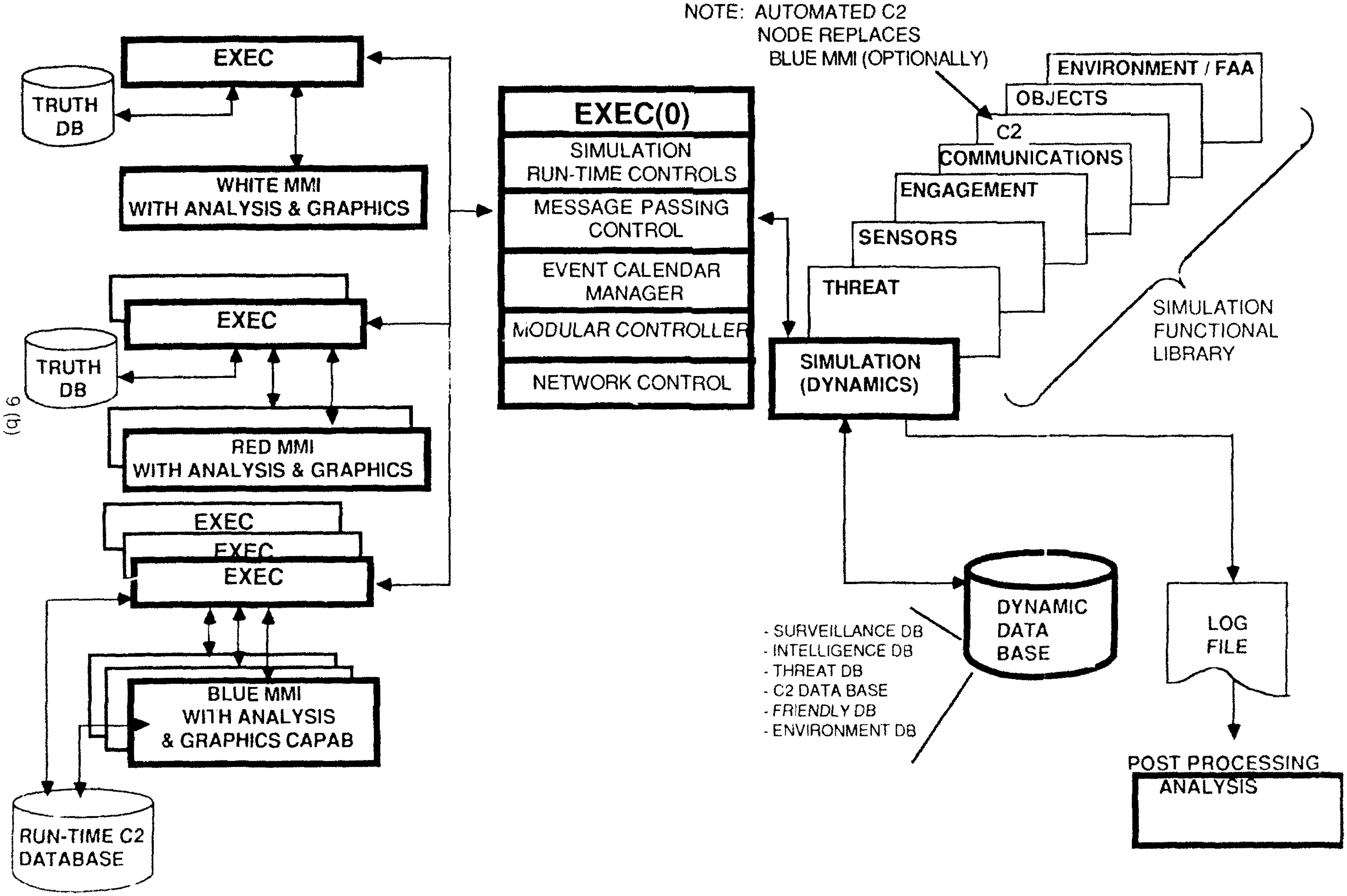
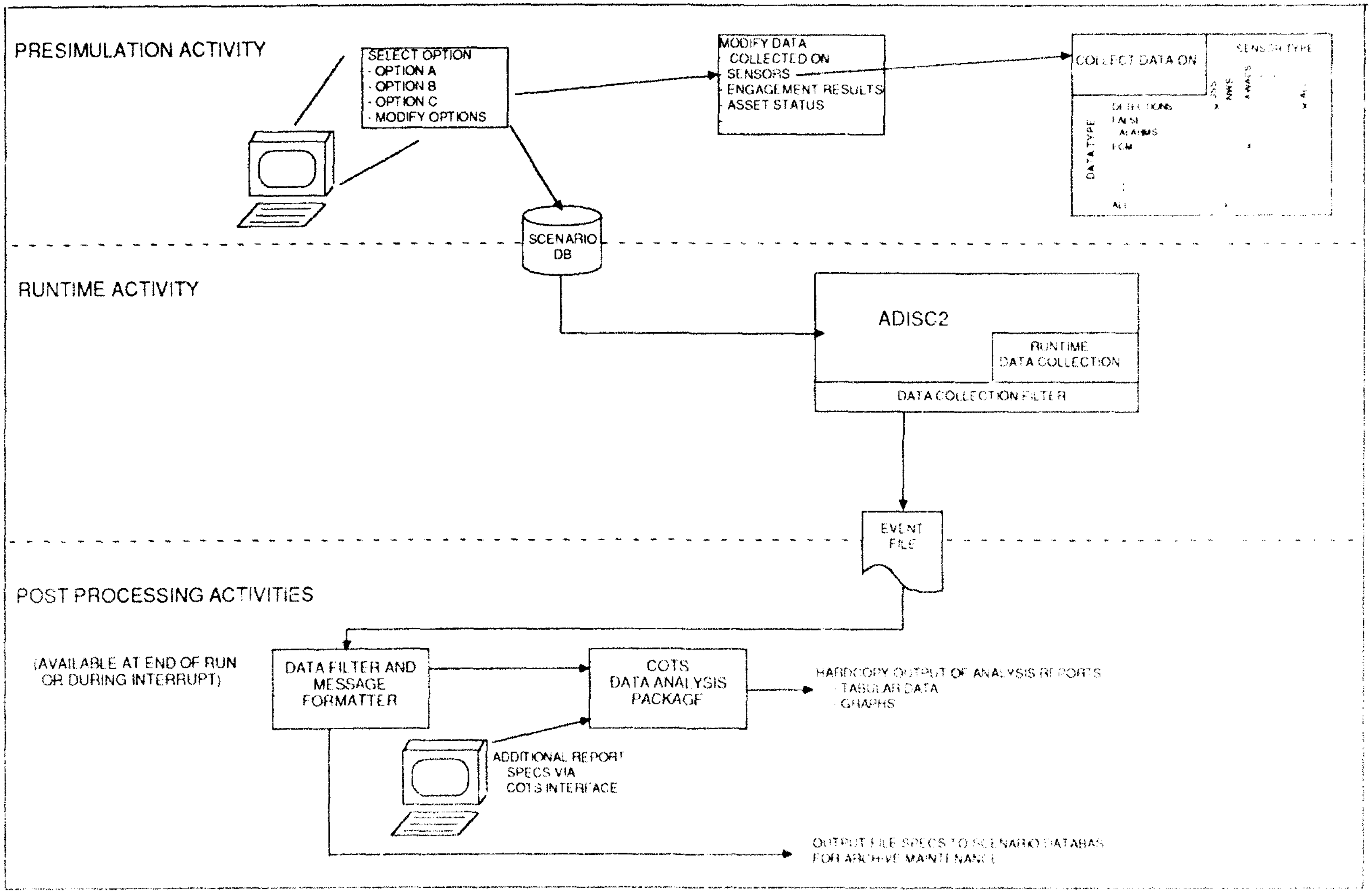


FIGURE 9 - POST PROCESSING DATA ANALYSIS CONCEPT



DATA COLLECTION SPECIFICATION APPROACH VIA
HIERARCHICAL MENUS

While there are many methods and products currently available that employ object-oriented design concepts, the more commonly used applications involve processes that keep track of every instance of an object whenever it changes function or operational environment. Many of the current approaches to object-oriented analysis and design require that each instance for a single object in the real world be defined and maintained as a separate object for each state in which it may exist (i.e., an aircraft object on the ground and the same aircraft flying are defined as two separate instances of the object and discrete sets of data are maintained for each instance). This multiplicity of object definitions in a volatile scenario imposes an unnecessary runtime overhead burden and often presents an unacceptable computational burden in strategic/tactical C^3 simulations.

Further, the limited flexibility of standard object-oriented approaches to respond rapidly to changes in object definition has been a problem for describing future systems. This has been a frequent, critical failure of object-oriented design when used in C^3 architecture evaluation and operational analysis problems.

The design of ADISC^2 utilizes standard object-oriented concepts and definitions but breaks with certain methods employed in most widely available COTS applications. This has also produced a slightly different vocabulary than is normally associated with object-oriented analysis and design.

For the current ADI applications an "object" is defined as something that is, or is capable of, being seen, touched (killed), or otherwise sensed. For purposes of ease of use and familiarity to our military users, we restrict the definition of an object to those things that correspond to objects in the real world, structured as they might be in a military logistics catalog or T.O. & E. description of a military unit.

Within the ADISC^2 SSE, an object is defined in a manner consistent with the Entity Attributes-Relational Attributes (EARA) model employed in most standard Object-Oriented Analysis applications. This allows logical structuring of the attributes of objects into sets unique to specific categories, classes, systems, sub-systems and components. Under this design, the table-building, archive management, and data access tools provided by the COTS relational database management software, are used in a pre-simulation interface mode to allow the user to create or employ and maintain tables of object attributes. The use of utilities provided by a relational database management system (DBMS) is significant in that the table-building utility offers virtually unlimited expansion of the breadth and depth of attribute descriptors for any object with a minimum of administrative overhead and database maintenance.

Tables may be created to incorporate and structure sets of attributes that describe objects. Tables incorporate attributes that uniquely identify objects, establish performance envelopes for objects, define the operational or mission environment of objects, or establish the basis for potential action/interaction among objects, systems, sub-systems, or components.

The logical layering used in the ADISC^2 design for object definition is:

- *Structural Frames within an environmental medium*
- Category
- Class
- Object
- Sub-system, by Type
- Component/Performance table

Within the SSE, an object is the lowest level construct that is discretely modeled in a given simulation application. The object is the lowest level that the user may define employment or deployment specifications.

From scenario to scenario, the definition of objects may vary, although the same data definition for an object and the same operational software may be used in different applications.

Objects are defined first as structural frames operating in one of several environmental media (spaceframes, airframes (aircraft or missiles), ground structures (fixed or mobile), ships/boats (surface or sub-surface vessels). In constructing a class of objects, the user must input those parametric values corresponding to a specific set of attributes common to all such structural frames. These attributes are defined as those necessary for the object to conform to the pertinent, fundamental laws of physics that apply to all bodies that exist or move in the environmental medium. Classes and objects within classes inherit the data formats for the set of all attributes defined for their environmental medium. Within a class, all objects will have the same parametric attribute values and carry the same sub-systems and components.

A "class", therefore, is a collection of objects that share the same set of descriptive attributes and that can perform the same set of functions. Performance attributes are only defined at the class or platform level and the initial attribute values are inherited by all sub-classes and objects/assets within a designated class.

Super-sets of classes may be created, called "categories". Categories include all objects within various classes having the same sub-sets of attributes, irrespective of the parametric values that define the different objects comprising each class. An example would be a category of "interceptor" that includes classes of F-15, F-16, F-14, F/A-18, etc. In the ADISC^2 design, the standard class definition has also been refined by introducing the concept of "structural frames", "systems/objects", "sub-systems" and "components" to allow the object-oriented taxonomy to adapt more easily to the strategic and tactical environment that the SSE currently serves.

In most standard object-oriented applications, the functions of each object (eg., VERB-like elements) are treated as attributes of the object and are, thereby, "hard-coded" as part of the object definition. This feature of standard object-oriented design was determined to be too restrictive to accommodate the requirements for flexibility in incorporating current and future performance envelopes of objects and it did not allow varying levels of fidelity to be represented in simulating object action and interaction for ADI evaluation.

The most significant departure in the ADISC^2 design from the standard object-oriented approach is in the separation of the functions of an object from its descriptive attributes. Unlike the standard object-oriented taxonomy described above, the functions of the object (the VERB-like structures) in the ADISC^2 are maintained separately with respect to object definitions and are stored as subroutines in a library. The design of the ADISC^2 SSE partitions object movement and operational, or mission, functions in the functional library. This library also is a repository of other activity-related subroutines and algorithms such as processes (complex PREDICATE-like structures) and utilities (PREDICATE MODIFIER-like and CONJUNCTION-like elements).

All motion options are invoked by a single controller (Executive) entry point in the SSE. This entry point is commonly referred to as the Objects Module. It is responsible for moving objects and knowing the location and state values of all structural frames, sub-systems and components operating in a given scenario execution. This promotes efficiency in the re-use of standard routines, inasmuch as the algorithm that moves an object from one waypoint to another along a great circle path is common to both Blue fighters and Red bombers; the functionality being based on the geometry and the physics, not on the other object attributes. Still, the performance of the movement of a bomber vs. a fighter will be varied according to the performance envelope of each object with respect to the attributes of each such as speed, range/fuel capacity, acceleration, climb and dive rates.

Table structures are also used to support the functional algorithms and provide input values or to serve as a repository for output values during execution. These tables, when populated with values, may perform ADVERB-like functions within the SSE. In this case, they are used to describe "how" a function is performed.

All performance attributes for objects have at least one initial/default value. For instance, if an object can change altitude in its operating environment, the algorithm that moves the object will assume instantaneous change when no further data regarding climb-rate is provided. However, if the database describing the object contains a table of values for climb, dive, acceleration, fuel consumption, etc., the appropriate algorithm will be invoked to limit performance according to the data available, using table-handling utilities built into the Simulation Support Environment. These functions are invoked by reference pointers to algorithms and related data tables via a linked-list of sub-systems and components interactively constructed by the user as an object type or class is defined.

Using these features of ADISC^2, it is possible to rapidly configure and employ a wide range of simulated objects, assign dynamic representation of varying degrees of fidelity to these objects and move them along pre-scripted routes and perform various actions and interactions with other objects.

ADISC^2 SOFTWARE INTEGRATION & RE-USE

The ADISC^2 design provides for "plug-in" modularity of functional software. Since the Simulation Executive reduces distributed simulation execution to a message-passing activity and provides the structure for standard message definition, it is possible to integrate legacy software with much less difficulty than with previous simulation approaches. While integration of legacy is never simple (and is sometimes impossible when the legacy is a continuous model and mathematical integration on a time variable is nct possible), the ADISC^2 design provides a structured approach that involves replacing the legacy main program with message calls to application subroutines coordinated by the Simulation Executive, structuring the input and output data required by each subroutine into standard message formats, and providing transformation utilities to resolve any incompatibilities in engineering units.

The design allows for legacy software, written in various high-level languages (Fortran, C, Pascal) to be integrated into composite executions under control of the Executive. Using this methodology, it has been possible to incorporate the functionality and/or original code from such models as those from the Research, Evaluation and Systems Analysis (RESA) System (from Naval Oceans Systems Center (NOSC) source), STRAT-DEFENDER (from AF/ESD source), MINI-MUF and OTH-B models (AF/Space Command source), and STRATC^{2}AM/SIMSTAR (provided by AF/S&A) in a matter of man-weeks.

The current version of ADISC^2 offers two fidelity options for radar sensor operation (simple fourth-root scaling or Signal-to-Noise ratio computation) and two types of OTH-B functionality (varying from Auroral Cap pertubation, solar effects and MINI-MUF calculations based on changes in the F-Layer of the Ionosphere, to simple "cookie-cutter", field-of-view representation).

The Communication Module presently offers the user three options for the simulation and performance evaluation of alternative architectures, systems or components and for evaluation of operational concepts for C^2 communcations: 1) Perfect communication in which all simulated links operate instantaneously without environmental perturbation, nuclear effects or jamming; 2) a network model incorporating link-level propogation routines from Air Force legacy code (STRATC^{2}AM/SIMSTAR) for propogation analysis across variable bandwidths, and including meteor-burst, tropo-scatter, jamming and nuclear effects; and 3) an RF propogation model called G-NET developed by GTE which has more coarse jamming and nuclear effects than the STRATC^{2}AM-based model.

Adding New Functions to the ADISC^2 SSE. Object sub-systems perform specialized functions such as sensing, sensor jamming/spoofing, communications transmission/reception/jamming, propulsion, guidance/navigation, defense, attack, tanking, etc. The operational tasks (ie., mission directives) of the object are stored and maintained by the Objects module although different mission activities may be executed under control of other modules (ie., Engagement, Sensor, etc.) For instance, the engagement cycle is maintained as an operational task for an interceptor object tasked to engage an assigned track. The Objects module is responsible for maintaining the data related to this and any other simultaneous engagement for this interceptor, initiating the engagement and updating parameters as status updates or new commands are received. The Engagement module is responsible for actually prosecuting the engagement by using data from the Objects module and occasionally updating the data stored for this task by the Objects module. Thus, the Objects module performs all the maintenance functions and the Engagement module performs the actual intercept and attack functions.

In its current state of development, not every aspect of the user interface can be managed by a single key stroke or by selecting a provided menu option. Perhaps this will never be desirable. The recommendation has been made to the RADC and ESD customer that the development of an intelligent interface to the Pre-simulation functions of archive management and configuration control would be beneficial. Such an interface, when fully developed, could allow the user to input analytical problems in English text, thereby invoking a system response. The response would present a potential solution space in terms of an SSE candidate hardware configuration and appropriate software applications and previously executed scenarios from the SSE archives. This interface might employ key-word indexes to the application software mapped to rules for potential problem solving. The system might eventually be able to learn new methods of defining the solution space from interpretation of historical user input specifications.

No matter how intelligent the interface becomes, however, the ADISC^2 user must be careful in the development of databases and scenarios for simulation of engagements. The artful employment of a complex tool will always be a challenge. Even though every attempt has been made to make the current prototyped ADI controls and system interface applications familiar and friendly to the user, there are few constraints imposed to keep mistakes from being made. This was a conscious design decision by the ADISC^2 developers.

Early in the program, it was decided to provide maximum flexibility to the user in definition of objects and configuration of the SSE so that users could easily simulate both current and future ADI systems and architectures. This decision was made as an alternative to restrictive editorial controls. This places the responsibility for proper definition and employment of objects and utilization of the capabilities of the ADISC^2 squarely on the shoulders of the user. The ADISC^2 user will find few controls within the SSE that have the potential for preventing the misdefinition of objects or for preventing the entering of parametric values that may be totally unreasonable.

In the hands of a knowledgeable user, the SSE can be configured to support high-level evaluations or extremely detailed analysis. However, both inexperienced and experienced users can make simple mistakes that may cause fatal execution errors or may seriously impact analytical results in ways that may not be readily apparent.

In one recent experience where a new object class within the "interceptor" category was being defined, the user placed a sensor sub-system normally found on an AWACs. The user then incorporated hundreds of these super-sensor interceptors into a scenario. When the execution began, the system was presented with thousands of detection messages from sensors with AWACs performance characteristics that were intended by the user only to lock-on to targets within air-to-air missile range of fighter aircraft.

The current ADISC^2 application is dimensioned to accommodate very large scenarios within the requirements of the customer for what are considered to be extremely rigorous air defense scenarios.

BACKGROUND

The objective of the Air Defense Initiative (ADI) is to develop and demonstrate the enabling technologies to conduct North American Air Defense against the cruise missile threat. The ADI is structured to make an informed full-scale development (FSD) decision in a time-frame consistent with the Strategic Defense Initiative (SDI) FSD decision. The concurrent development of C^2 with Surveillance/Engagement capabilities is essential to present total system capability for FSD. ADI Surveillance and Engagement technologies are being vigorously pursued in other programs. This effort models those capabilities and provides the context for concurrent development of $ADISC^2$ functions.

PURPOSE OF THE PROGRAM

The objective of the $ADISC^2$ is to provide the capability to develop the technology to demonstrate Command and Control functions consistent with the development of other ADI capabilities. This has been accomplished by the modeling of current and advanced systems in the areas of Surveillance, Engagement and Communications.

The first priority of this effort, in Phase I, was to model the threat scenarios and sensor systems that constitute baseline surveillance and identification capabilities for Air Defense, on which alternative advanced technology systems will be built. Consequently, the initial emphasis of this effort was to model the Threat and Sensor systems required to provide Surveillance and Identification functional capabilities. Phase II activities involved the implementation and incorporation of the Engagement, Air Traffic and Communication modules as designed in the Phase I activity. Engineering Changes to the base contract resulted in the development of an Automated Command & Control and an Analytical Module to the Simulation Support Environment of the $ADISC^2$ to support the evaluation of alternative ADI architectures without man-in-the-loop involvement.

Man-in-the-loop decision-making is stressed for many ADI Command and Control concepts. This effort provides the baseline for determining the degree of automation necessary to support the decision-making process. Command and Control of Air Defense requires coordination and direction of the activities of many different Air Defense elements, consistent with the time constraints imposed by the threat and necessary response times.

The technology to demonstrate C^2 functions will be developed and implemented using $ADISC^2$ simulation, thereby insuring interoperability within a consistent Air Defense simulation environment. These C^2 functions will provide support to the man-in-the-loop as he executes the functions of Air Defense Command and Control. This, in turn, provides a unique opportunity to concurrently develop C^2 functions with the development of advanced technology Air Defense systems.

The $ADISC^2$ also provides the means for interfacing to, and interacting with, other facilities and systems in order to demonstrate, for example, the potential of distributed C^2 operations by placing the operational user in the developmental/evaluation loop. The man-in-the-loop evaluation process provides the capability for evaluation and analysis of C^2 technology for the ADI as well as providing a vehicle for operational-user involvement within the C^2 development process.

PROJECT DETAILS

The ADISC^2 design and development activities were performed in two phases, each comprised of several tasks.

PHASE I included the following:

PHASE I; Task I - Define the concept of ADISC^2 in the context of the North American Air Defense Environment (NAADE). The NAADE covers all of North America and the points of origin of all threats to that continent. Within the NAADE there exist variable factors such as the location of the Command and Control nodes, strategic targets of relative importance (depending on the Threat objective and mission), commercial and military air traffic, force generation, deployment/employment of forces, US/adversary capabilities, and rules of engagement that influence potential levels of conflict. The concept of a realistic ADISC^2 had to reflect and provide for operations within a simulation of this environment. The ADISC^2 includes the ability to realistically represent scenarios within which various technologies may be demonstrated and their performance characteristics graphically displayed and evaluated.

PHASE I; Task II - Design the simulation hardware and software architecture to include the capability to generate realistic threat scenarios based on Government-Furnished Information (GFI) and the modeling of appropriate operating components (eg., sensors, defensive assets, and engagement systems) within the NAADE. The data necessary to support these systems was incorporated into a relational data structure using a Commercial Off-the-Shelf (COTS) Database Management System (DBMS) called INGRES. INGRES was selected as a result of a trade study that included other COTS DBMS products such as ORACLE and Sun UNIFY. INGRES offered two-and-a-half times the rate of retrieval of its closest competitor and offers nearly unlimited expandability. The design also incorporated modular components, included the specification of hardware and software requirements, and addressed the performance requirements for supporting real-time and faster-than-real-time simulation executions. The design includes support elements for simulation experiments, data collection, data analysis, multi-user and user-friendly operation and future interoperability.

The design was developed as an "open architecture" solution capable of residing on most commercial hardware and able to be supported by the hardware suite at the RADC Command and Control Technology Laboratory (C^2TL) and the Mobile Laboratory (ML). The design was not constrained, however, to the current hardware configuration within either of these facilities. Sub-tasks were performed including the conduct of a parametric analysis of the software to determine the impact of increases in number, granularity and fidelity of threats, sensors and other simulation elements on the speed of scenario execution, requirements for data storage and retrieval, computer architecture and hardware parameters, terminal/display capabilities, and operating system requirements. The results of this analysis were interpreted and recommendations were formulated in the form of a hardware architecture design and a software-to-hardware mapping. An architecture of multiple Sun 4 workstations, connected by a high-speed local area network and supported by secure and non-secure virtual storage devices, printer and color graphic plotter was submifted.

Other studies were conducted in support of the ADISC^2 design including the assessment of impact of adding an Analytical Software module and a Command and Control module. These recommended enhancements were incorporated as a result of modification to the origional contract during Phase II.

Design activities performed in Phase I also included the requirements definition and development of detailed design documentation for Scenario Generator, Simulation Controller, Display Software, and Functional Modules of Threats, Sensors, Engagement, Communications, and Air-Traffic.

PHASE I; Task III - Acquisition and coding of the Software Architecture required to implement the simulation-support architecture designed in Task II (ie., Scenario Generator, Simulation Controller, and Terminal/Display Software) was completed.

PHASE I; Task IV - Implementation of the Threat Module was completed.

PHASE I; Task V - Implementation of the Sensors Module was completed.

PHASE I; Task VI - A demonstration of capability developed in PHASE I was performed successfully at the contractor's facility.

PHASE II tasks included:

PHASE II; Task I - Implementation of the Engagement module to allow control of the engagement forces (defensive weapons) and to perform simulated engagement between defensive forces and hostile penetrator forces.

PHASE II; Task II - Implementation of the Communications Module to allow simulation of message loadings, throughput, link analysis and connectivity among simulated elements of the NAADE.

PHASE II; Task III - Implementation of the Air Traffic Module was completed in Phase I ahead of schedule. With the object-oriented design solution developed for ADISC^2, it was possible to create commercial, military and strategic aircraft objects that were interactively controlled as well as air-traffic objects used as background to the simulation of threats and engagement elements.

PHASE II; Task IV - Preliminary Acceptance Testing was conducted on all system-level software at the contractor's facility. Over 270,000 lines of code was developed and tested, in addition to the 380,000 lines of COTS and OTS software integrated and delivered. This was accomplished with an average commitment of twelve software engineers over a period of about 36 months.

Rapid prototyping of modules and displays required development of new and innovative software test methods. Modular testing at the unit level was performed with Government witness and involvement routinely throughout the development activity. Emphasis was placed on "doing the right things right, and doing them right the first time". Any errors noted at the unit level were corrected immediately. Once approved at this level, the code modules were placed under configuration management and control of the Martin Marietta Quality Assurance staff who were assigned the responsibility for the program.

Integration testing was performed at the end of the Phase II effort to establish the ability of the simulation support environment and the software applications to satisfy the contract requirements and to provide evidence that the approved unit modules were properly integrated. During the AcceptanceTests and the Operations & Maintenance (O&M) period that followed over the next seven calendar months, a total of 94 Software Problem/Correction Reports (SPCRs) were processed. Most were minor in nature and were immediately corrected. Some were the result of "bugs" in the operating system software supplied by SUN Microsystems. These issues and related "Lessons Learned" are discussed further in later sections of this report.

PHASE II; Task V - Installation of the ADISC^2 software was completed at the RADC C^{2}TL in April, 1990.

PHASE II; Task VI - Final Acceptance Test of the complete ADISC^2 was performed at RADC in April. 1990.

PHASE II; Task VII - Operation and Maintenance Support was provided for a period of ninety days from government acceptance of the software. Due to additional development activities performed under modifications to the origional contract, the actual performance of O&M activities extended over a seven-month period, from April through October, 1990.

PHASE II; Task VIII - Associate Contractor Relationships were established to insure compatibility of the ADISC^2 software with other development efforts through their respective contractors, including:

- Survivable C^2 Center Technology Demonstrator
- ADI Technical Evaluation Facility
- SDI National Test Bed/ National Test Facility
- CONUS Region Operations Control Center
- Northeast Sector Operations Control Center

Due to the open-system and message-passing features of the ADISC^2 design, the capability to interface with external JSS sensor systems became a simple matter of developing a single translation/interface program. This program was written to assure that messages passed between real and simulated points of origin were properly formatted for use by either system.

It was determined by RADC and ESD that the hardware/software architecture developed for ADISC^2 could economically be modified with the addition of an Automated C^2 Module and an Analytical Module and satisfy the requirements for ADI architecture evaluation.

First Contract Modification; At the end of Phase I, an Engineering Change Proposal (ECP) was submitted that allowed Martin Marietta to acquire for the Government a SUN-4/280 file server system and three (3) SUN 4/260 graphic workstations. These machines, together with secure and non-secure storage devices, printer and color plotter provided a hardware configuration capable of supporting ADISC^2 development and operation.

Second Contract Modification; During Phase II, a second ECP was adopted that modified the original version of the ADISC^2 software by adding an automated Command & Control Module and the Analytical Module in order to support the planned evaluation of various ADI Architecture concepts by ESD. Additional workstations, printers, plotters and storage devices were acquired and delivered to RADC and ESD. This ECP extended the original contract completion date by approximately six (6) months and provided additional Operation & Maintenance (O&M) contract labor hours to RADC and ESD.

In addition, RADC provided a legacy communications model from GTE known as G-NET. This model is an RF propagation model. RADC directed the incorporation of the G-NET software into the SSE as a user-selectable alternative to the STRATC^{2}AM/SIMSTAR model delivered under the original agreement. This effort demonstrated the ability of the SSE to rapidly and faithfully employ legacy software to support ADI C^2 development and evaluation.

Third Contract Modification; The contract was again extended and amended to allow Martin Marietta to provide additional training and O&M effort to ESD.

ADISC^2 OPERATIONAL MODES

The ADISC^2 design is best described in terms of its three operational modes; Pre-simulation, Runtime and Post-processing Analysis.

ADISC^2 Pre-Simulation Functions. The ADISC^2 provides the user with an interactive means for generating, archiving, editing and maintaining scenarios and for configuring the hardware/software architecture in order to accomplish simulation objectives. Pre-Simulation Operations consist of the functions required to support the simulation such as creating and maintaining the databases, system security, system software configuration, system hardware configuration, scenario generation, and archive maintenance.

There are two types of screens for the user to interface with the ADISC^2 system; the database menu type and user interactive graphics type. Database menu type screens have selection options displayed at the bottom of the screen. Selection of a pre-simulation function is accomplished by positioning the cursor on the information desired and entering selection commands through use of the appropriate function key identified from the menu display. Interactive graphic screens have menus with controls or icons on the map display which are activated using the mouse. Selection is made by positioning the arrow on a control or icon and making selection by mouse control. Either type of display may request input from the user; the user will have prompts provided to explain what input is required.

There are four major functiona' areas within the ADISC^2 Pre-Simulation mode:

- **Administration**, aimed toward operators interested in ADISC^2 system support and asset allocation. Administration functions incorporate Database Maintenance and Database Utilities elements as well as a scenario Merge and Preview capability;

- **Prototyping**, for users who are interested in the effects of new equipment and technology on the air defense environment.

- **Scenario Generation**, of primary interest to users of an operations orientation; and

- **Run Configuration**, to assist users in developing configuration specifications fo the hardware/software architecture and to input run control parameters, such as beginning time of the scenario, real-time to simulation clock ratio, etc.

Administration Functions support the configuration and application sofware management and maintenance activities required by ADISC^2, including Simulation Executive configuration, hardware configuration, run configuration, software configuration, utility configuration and database maintenance. The Simulation Executive configuration function provides an interface for building and maintaining executive file configurations and simulation input files. The hardware configuration function provides an interface for building network node configurations. Run configuration defines simulation run configurations with pointers to database and scenario, simulation timing, utility files selection, software module selections, and replay information. The software configurator interface maintains a list of optional simulation modules. Utility configuration provides an interface for creating input files for the simulation utility preprocessing functions and for maintaining the utility output file names.

Database Maintenance Unit provides a user interface to INGRES databases for the purpose of maintaining databases, scenarios and ADISC^2 simulation configurations. It provides several automated maintenance features for the various scenario databases that are available within the SSE. It will perform database checkpoints at the database administrator's request, run a "modify" script to remove storage overflow in the larger database tables, and run an "optimizer" script to generate statistics for the INGRES optimizer. Additionally, this unit displays information about the scenarios in the databases, allows users to create new scenarios, delete scenarios, archive scenarios, retrieve archived scenarios, and update the scenario library information.

Database Utilities Unit provides the function of database and scenario selection. This unit opens a scenario database and sets a global pointer to identify the selected scenario. Also, this unit maintains consistency between the information from the scenario databases and the database library management indexes, such as updates to the last date that a scenario was accessed through the Scenario Generator.

Preview & Merge Unit provides a capability to select subsets of scenario data for a "preview" or new scenario. The merge function allows the user to bring together elements of 1 to 6 other scenarios and merge them into a new scenario. The preview function allows the user to select a subset of elements from one existing scenario and incorporate them into a "preview" scenario. This is available in a 2-node configuration for simulation execution.

Network Mail Service provides the operator with the ability to construct unformatted or pre-formatted text messages for routing to system nodes/users. This may facilitate passing of "situations" or instructions to users at remote nodes, informing them of times of scheduled simulation activities, defining their role as test subjects, providing information as to their force structure or deployment at the beginning of a simulation, defining enemy situation associated with a particular scenario. Other messages may be routed or broadcast that relate to the operational status of the system, scheduled maintenance activities, etc.

Prototyping Functions provide a user interface to the INGRES databases for the purpose of prototyping simulation elements: object classes (platforms), sub-systems, and performance tables. These units provide an interface to the Scenario Archive Database.

The ADISC^2 design incorporates an extensive Rapid Prototyping capability that allows the user to rapidly establish new classes of objects, incorporate new functions or dynamic utilities, and generate new animated graphics and displays that will be used during simulation execution. This design feature is intended to provide the greatest degree of flexibility possible in generating simulation primitives or "building blocks" that can be rapidly mixed-and-matched to build a wide variety of ADI scenarios.

Users invoking the Rapid Prototyping options may access the INGRES Database Manager through ADISC^2 menus in order to add new types of objects. The ADISC^2 object-oriented design allows such objects to be defined in a modular fashion, beginning with the platform description within an operational medium (ie, ground-fixed, ground-mobile, air-frame, sea-surface, sea-submarine, space-platform) and proceeding through the steps of equipping the platform with various surveillance, weapons, propulsion, and communications sub-systems and definition of the set of parametric values that establish the object's "performance envelope". Once defined, the object can be incorporated into an ADISC^2 Scenario by receiving initial deployment under the Scenario Generator/Force Structuring options.

Objects are allowed to participate in the simulation via dynamic utilities. These utilities are subroutines that may be called by the Simulation using messages passed by the Executive. Some of these subroutines move objects from way-point to way-point in pre-defined missions or to new way-points injected interactively during execution. There may be various algorithms used to achieve movement, from Great-Circle routines, to Curve-fitting trajectories, to six-degree-of-freedom calculations. Other utilities allow the object to perform simulated operations such as radar surveillance. These utilities are classified and placed in a library in INGRES. New phenomenology may be included as entries in the library. Maps or "logical pointers" are built to objects via their assigned sub-systems to ensure that appropriate utilities are matched to the performance envelope of each object. This insures that submarine objects don't fly and air-frame objects don't swim underwater unless such capabilities are desired by the user.

In this manner, considerable flexibility is given to the user of ADISC^2 in being able to respond to emerging technologies or new C^2 concepts in air defense. It is quite simple to simulate performance of new systems of sensor objects mapped to standard dynamic utilities such as formulas of radar phenomenology as found in Skolnik's Radar

Handbook[2] or Introduction to Radar[3]. In fact, ADISC^2 offers several ways that a user may create an object or define variations to a class, or type, of object.

The user may define a new object by incorporation of new software capable of simulating functions or operations not performed by objects in the ADISC^2 SSE. Having added this software to the library of functional subroutines, the user may define new sub-systems that perform the newly-added function. This definition may include the creation and population of tables that define the performance envelope of an object performing this function or that provide parameter values necessary for execution of the functional subroutine. Once these steps have been performed and the sub-systems have been integrated and archived into the SSE, the user may create new object categories, classes and types by mixing and matching available object sub-systems on various structures (ie., the user may add a weapon sub-system to an existing E-3 aircraft in order to simulate a "missileer" air defense platform). In addition, the user may create new objects simply by changing one or more parameter values that produces a new version of an existing object (ie., the user allows an interceptor to fly Mach 5, thereby creating a future class or type of an interceptor).

Recognizing that such capability for rapid object generation requires similar flexibility in creation/modification and/or rapid prototyping of graphics and displays, the ADISC^2 design incorporates applications written under Template/Blox that allow users to rapidly create new formats and tie them to simulation executions. The standard message definitions used in the ADISC^2 design provide the user with the opportunity to collect and utilize data from the message stream at any desired collection interval. Data may be used to support the analytical purposes of the user and may also be used to populate tabular or graphical displays that the user may define through the MMI.

Flexibility in creation of tabular displays of data, or in parametric data displayed as text notation on graphics displays, is therefore assured. The MMI applications developed for ADISC2, allow definition of tabular displays by the user with "pointers" to the appropriate data, generated either from database access or by system-generated output. Changes to the tables are posted via messages passed by the Executive through designated display graphics "sockets" or "mailboxes".

Among current capabilities provided by ADISC^2 are: ability to change display colors, ability to re-format any display via alternative locations and dimensions of pop-up windows or pull-down menus, ability to add rows or columns to tables that are dynamically posted by the simulation during execution, the ability to rapidly define or change format and content of text displays used as map overlays and as interactive text on icon interrogation, and the ability to rapidly generate new map overlays displaying data from the INGRES database or from the dynamic data sets maintained during execution.

The following software units are included in Prototyping: Database Prototyping Controls, Object Class Definitions, Performance Table Editing, Subsystem Definitions.

[2] Skolnik, Merril, Radar Handbook; Mcgraw-Hill Book Company. New York, N.Y.; 1970

[3] Skolnik, Merril, Radar Design Principles; Mcgraw-Hill Book Company, New York, N.Y.; 1960

Database Prototyping Controls Unit provides the top level control for database prototyping. The "Database Prototyping Main Menu" passes control to the lower level units.

Object Class Definitions Unit provides the interface to prototyping object classes. Object classes, also referred to as object structures, are the basic building blocks of the objects to be used in simulation executions. Classes are defined by side (red, blue, white), operational medium of the structure (air, space, fixed ground, mobile ground, naval, submarine), and category (bomber, airborne sensor, air base, ground sensor, etc.) These are deployed in scenarios to become simulation objects that detect, fly, drop bombs, etc. Through this unit, the user may modify the characteristics of existing classes, create new classes, and delete obsolete classes.

Performance Table Editing Unit provides the interface to defining performance tables and environmental effects tables. Performance tables provide unique performance characteristics to the object classes & sub-systems that use them. Performance tables are definable in data sub-tables for such characteristics as radar cross-section, fuel utilization, acceleration rate, etc. Sub-tables of a given table are a sub-set of its rows. Any number of object classes may use any given performance sub-table. Through this unit, the user may modify the characteristics of existing performance sub-tables, create new sub-tables, and delete obsolete sub-tables from the database.

Sub-system Definitions Unit provides the interface for the prototyping of object sub-systems. Sub-systems carried on object platforms provide unique dynamic and operational capabilities to the object classes. Sub-systems are defined by type such as sensor, radar jammer, communications suite, armament, etc. Any number of object classes may contain a given sub-system. *Through this unit, the user may modify* the characteristics of existing sub-systems, create new sub-systems, and delete obsolete sub-systems from the database.

Scenario Generator provides a user interface to the INGRES Scenario Databases for the purpose of creating and modifying ADISC2 scenarios. Deployment of simulation objects, threat attack missions, defense planning, and pre-scripted environment data is displayed and manipulated through this software. The following software units are incorporated in the Scenario Generator Module: Aircraft Deployment, Blue Planning, C^2/Base Deployment, Communications Planning, Data Collection Pre-Processor, Navy Deployment, Pre-Simulation MMI Support, Red Planning, Route Planning, Scenario Generator Controls, Scenario Generator Utilities, and Sensor Deployment.

The design of the ADISC^2 Scenario Generator is intended to allow the Operational Analyst/Military Commander/ROCC or SOCC Operator to interface with the system in a manner that allows the construction of ADI scenarios consistent with the operational functions of each user. For the Systems Analyst/System Engineer interested in *evaluation of C^2 concepts or technologies, this orientation is intended to ensure that a* high level of accuracy and realism is maintained in the process of scenario generation.

The Scenario Generator design concept is straightforward and based on a relatively simple objective; the user builds a scenario using operationally familiar interfaces and employing the same activities and techniques as might be used by those involved in Red mission and Blue defense system planning. Because many assumptions and simplifications had to be made in an attempt to absolutely minimize the level of effort required to build a Red threat scenario, realism in such activities as target selection, load-out and weapon target pairing were considerably compromised and simplified. However, the simple rules and algorithms currently incorporated provide "hooks" for future enhancement while providing a level of realism seldom associated with building computer simulation scenarios.

There were two primary purposes of the prototyped planning interfaces provided with the initial delivery of the ADISC^2; 1) To demonstrate the capability of the SSE to rapid prototype useful interactive graphic C^2 displays, and 2) to provide ADISC^2 user with a means of rapidly configuring a baseline threat scenario and for developing and employing numerous excursions to that baseline in support of C^2 evaluations.

In any case, the user may always elect the alternative menu interfaces to the DBMS to define detailed routes for individual objects, map individual threats to selected individual targets (with up to 3,000 possible targets to address), and perform individual load-outs for every carrier rather than utilize the more rapid, but simplified, system-assisted options available in the design. Therefore, if extreme accuracy is desired in RED SIOP-level planning, the design does not preclude the achievement of that objective.

Aircraft Deployment Unit provides the interface to creating aircraft objects and deploying aircraft in a scenario. This unit allows the user to create new objects and modify initial conditions for existing mobile air objects.

Blue Planning Unit provides the control for blue planning functions. Menus for access to deployment and assignment of blue forces in a scenario are provided. Additionally, access to defense and command and control planning is included.

C^2 Base Deployment Unit provides the interface to deploying C^2 sites, Surface-to-Air Missile (SAM) sites, and air bases in a scenario. This unit allows the user to create new objects and modify initial conditions for existing objects of these types.

Communications Planning Unit provides the interface for establishing communications links and routes for a scenario. The communications links are established between existing objects that have common or interoperable communication sub-systems. Communications routes originate with one object and connect 1 to 3 links to other objects and consolidate them into a single route.

Data Collection Pre-Processor Unit supports the interface for establishing data collection options for a scenario. This unit allows the user to turn data collection on & off, and define message, class, & object collection configurations.

Navy Deployment Unit provides the interface to deploying naval forces in a scenario. This unit allows the user to create new objects and modify initial conditions for existing objects of naval surface and submarine classes.

Pre-Simulation MMI Support Unit establishes the database interfaces for the pre-simulation MMI programs (Red Mission Planner, Blue Mission Planner, and Platter Program.) This unit consists of subroutines called by the MMI programs to provide retrieves and updates to the scenario databases.

Red Planning Unit provides the control for Red planning functions Menus for access to deployment and assignment of red forces in a scenario are provided. Red Planning is divided into relevant tasks of force structuring, air defenses route/mission planning, target selection, weapon/target pairing, and attack timing synchronization.

Route Editing Unit provides the interface to editing and maintaining routes for a scenario in a tabular format. This unit provides the interfaces for defining routes for Red and blue mobile forces.

Scenario Generator Controls Unit provides the top level control for scenario generation. The top level menu that gives control to the lower level units is included. Environment definition functions are within this unit.

Scenario Generator Utilities Unit provides utilities used throughout pre-simulation for forms and database manipulation. This unit includes conversion functions, string manipulation functions, and utilities frequently accessed by display routines for various forms actions.

Sensor Deployment Unit provides the interface to deploying blue surveillance forces in a scenario. This unit allows the user to create new objects and modify initial conditions for existing objects with surveillance capabilities.

Run Configuration assists users in developing specifications of hardware and software configuration data required for simulation execution in an interactive manner. This software module consists of interfaces and control software that provides the user with access to the Simulation Executive run specification input stream for user-definable parameters such as "simulation start time". Additionally, through this interface, the user may indicate whether a simulation execution is to be a batch or interactive mode, time-stepped or event-driven, a single or multiple node configuration, and the scenario elements to be included in a given run.

In Pre-Simulation mode, the user accesses the ADISC^2 environment by a main menu similar to that shown in Figure 10. Figure 11 presents a graphical representation of the ADISC^2 design and the active functional areas involved in the Pre-Simulation mode. Figure 12 presents a series of activity flows to illustrate the Pre-Simulation Processing of ADISC^2.

ADISC2 During Initialization.

During the initialization step, the Simulation Controller performs the function of obtaining the initial parametric data values from the INGRES DBMS, and distributing the required data to each of the nodes based upon the configuration requirements and specifications. All calculations required for execution that can be pre-computed are performed by the various sub-modules. Outputs from these pre-simulation computations that are related to scheduled events are passed into the event stack. During this activity, the user is notified when various steps are performed as the process leading to the completion of the initialization routine is executed.

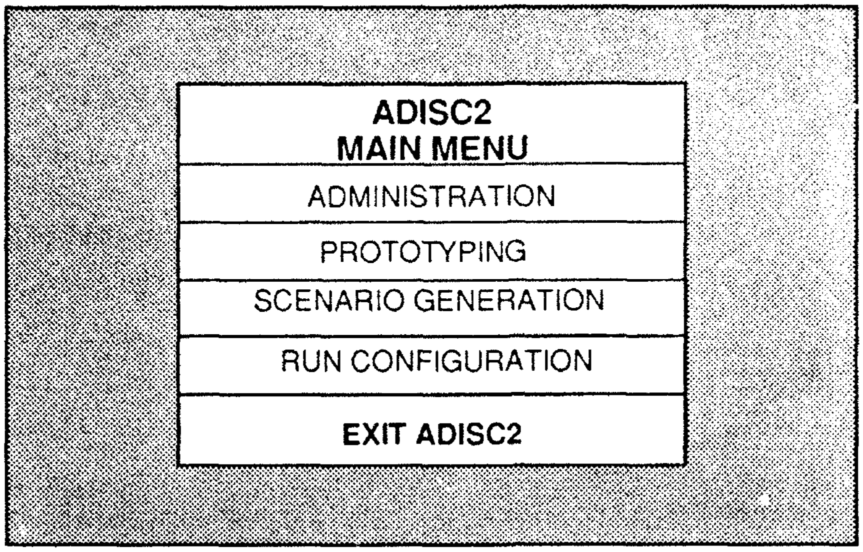


Figure 10: THE MAIN MENU PROVIDES A SINGLE POINT OF ENTRY FOR CONVENIENT USER ACCESS TO THE SSE

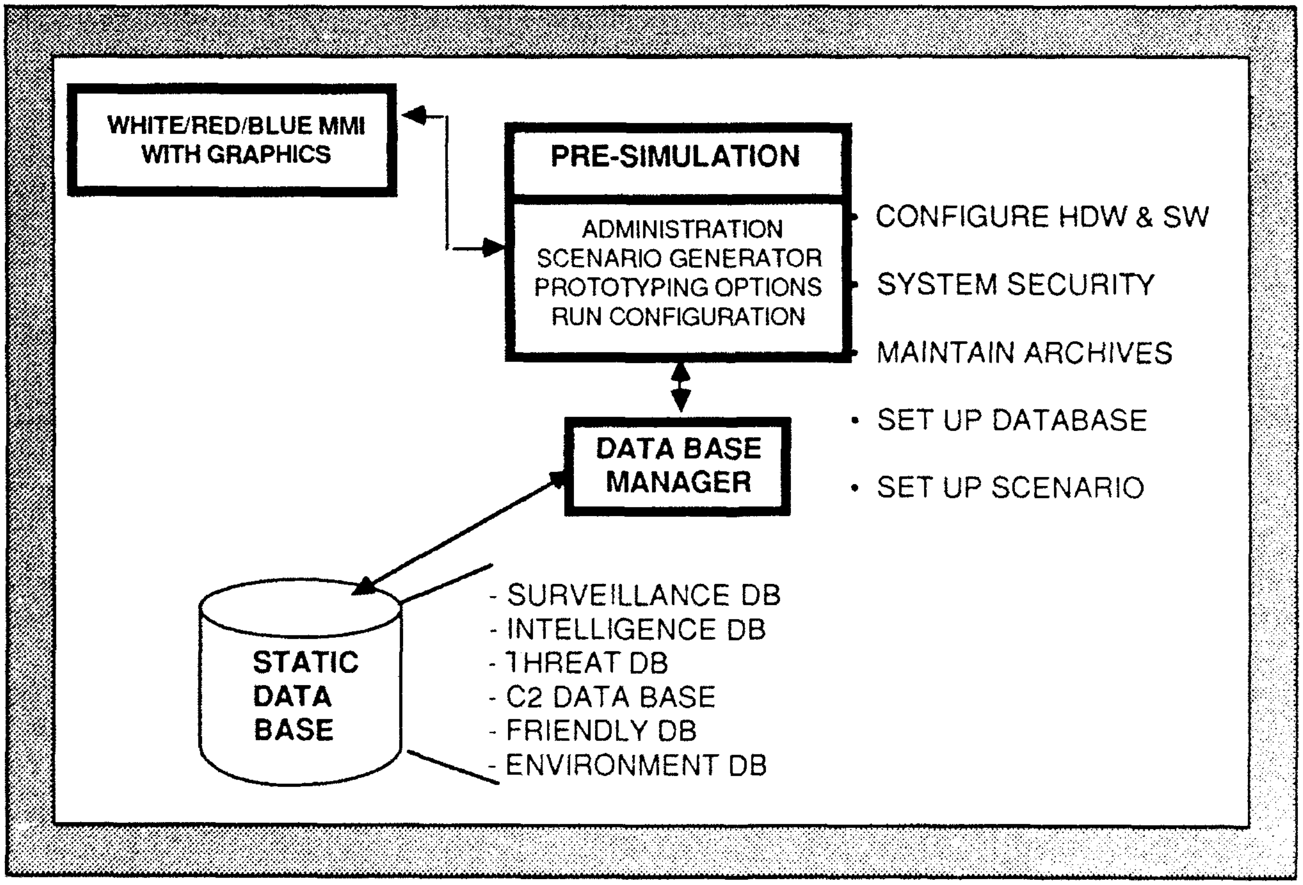


FIGURE 11 - ADISC2 PRE-SIMULATION FUNCTIONAL DESIGN CONCEPT

FIGURE 12 - ADISC2 PROCESSING THREAD/PRESIMULATION

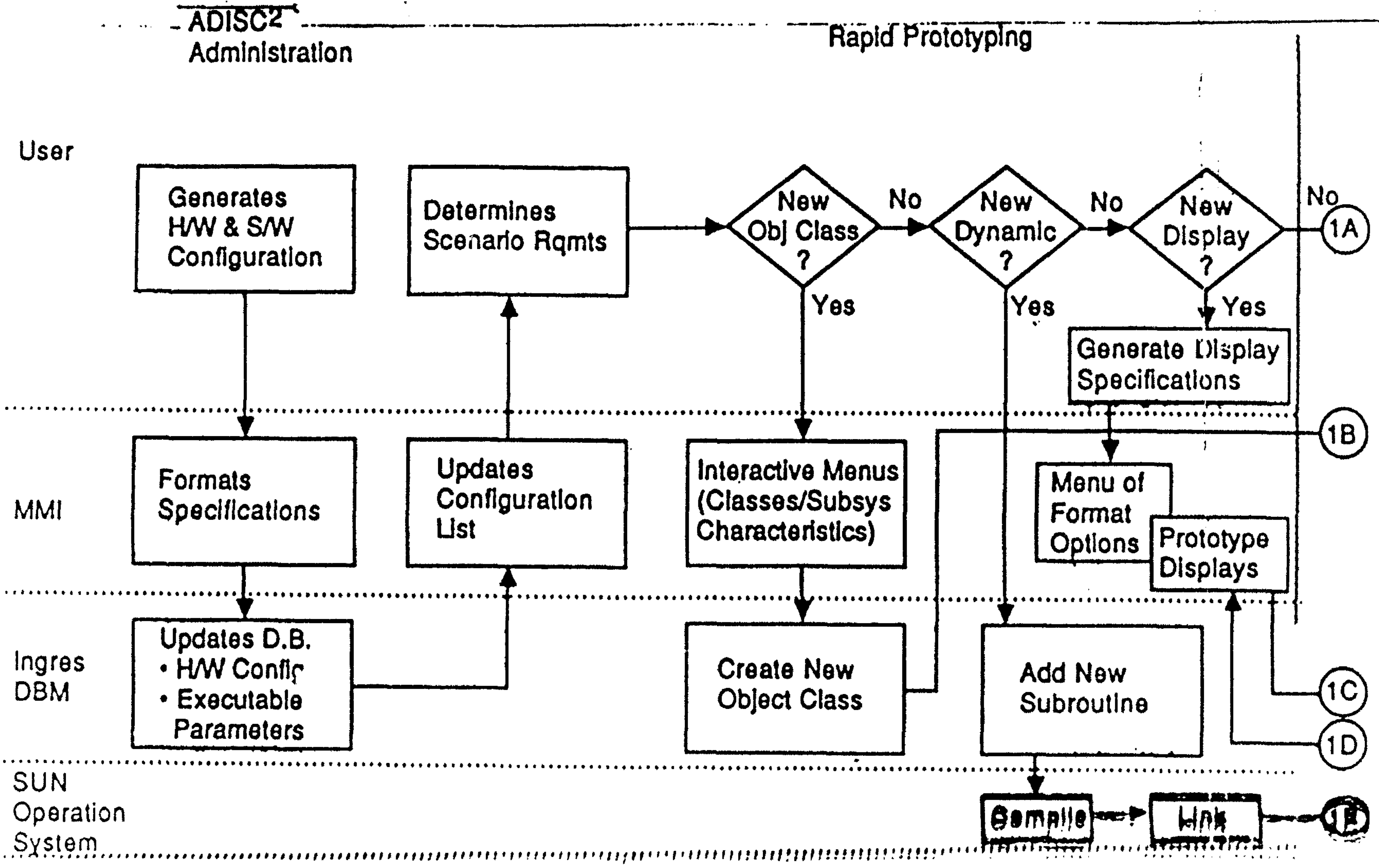


FIGURE 12 - ADISC2 PROCESSING THREAD/PRESIMULATION (CONT)

Scenario Generation/Selection

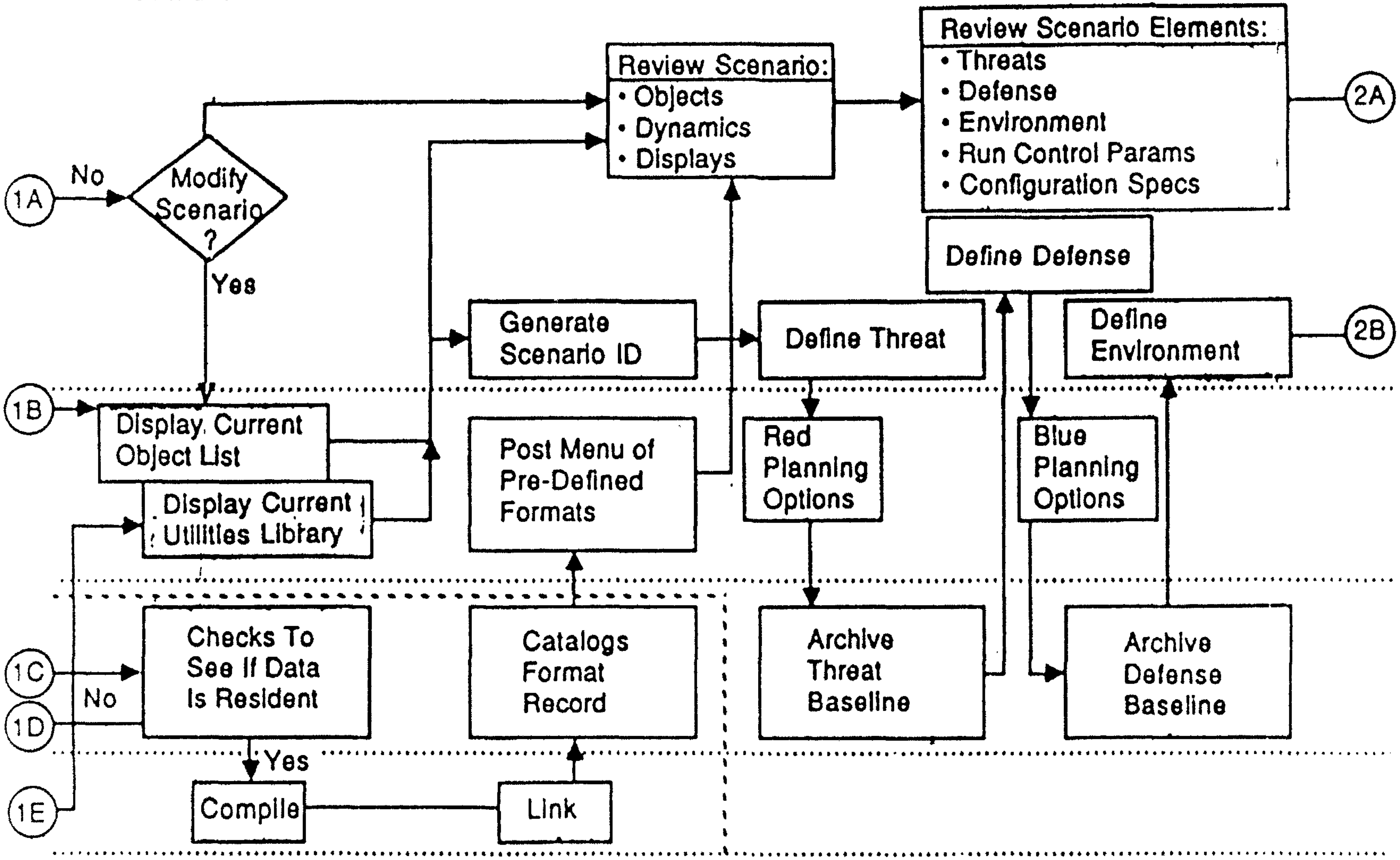


FIGURE 12 - ADISC2 PROCESSING THREAD/PRESIMULATION (CONT)

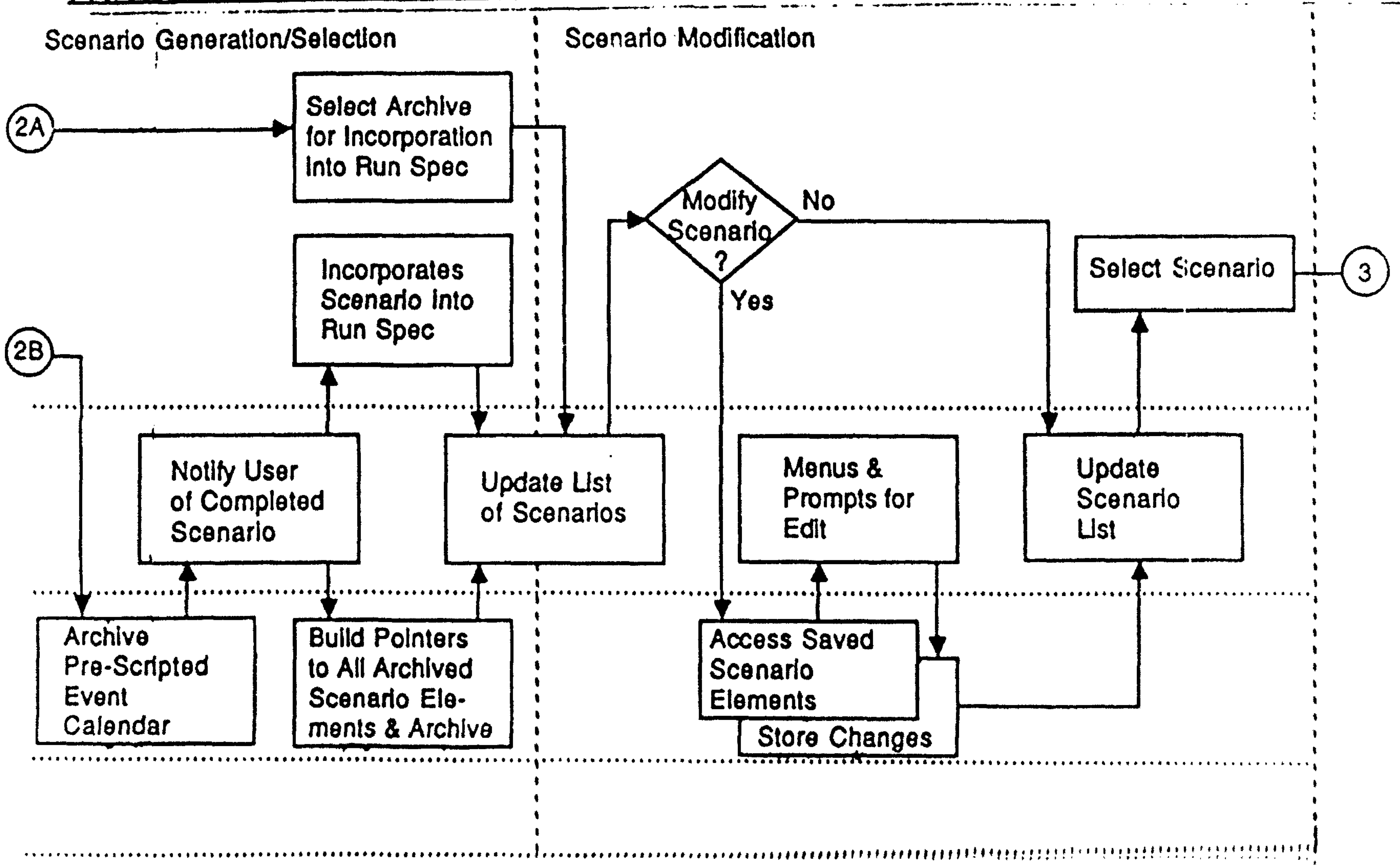
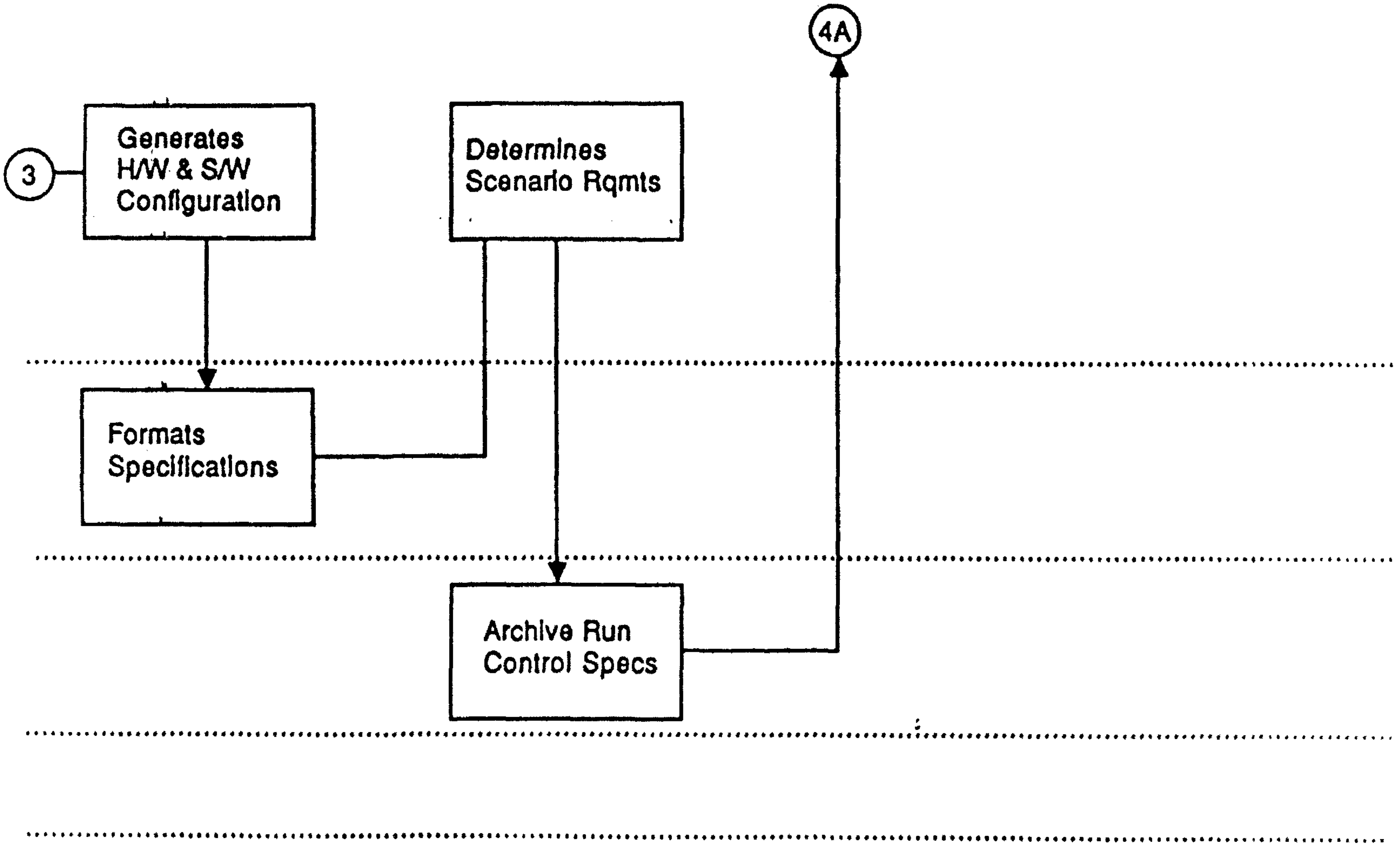

Scenario Generation/Selection
Scenario Modification
2A
Select Archive for Incorporation Into Run Spec
2B
Incorporates Scenario Into Run Spec
Notify User of Completed Scenario
Update List of Scenarios
Modify Scenario ?
No
Yes
Select Scenario
3
Menus & Prompts for Edit
Update Scenario List
Archive Pre-Scripted Event Calendar
Build Pointers to All Archived Scenario Elements & Archive
Access Saved Scenario Elements
Store Changes

FIGURE 12 - ADISC2 PROCESSING THREAD/PRESIMULATION (CONT)

Run Control Specification


3
Generates H/W & S/W Configuration
Determines Scenario Rqmts
4A
Formats Specifications
Archive Run Control Specs

The purpose of the initialization module is to provide a bridge between the scenario data stored in the INGRES data base and the simulation. This is effected by retrieving the data from the data base and initializing the simulation working variables with this data. The simulation working variables are all stored in the simulation common blocks. These common blocks are documented in the common block section of the <u>ADISC^2 Programmer's Manual</u>.

A secondary requirement of the initialization process is to initialize some working variables in these common blocks that are not initialized from the data base. This includes initializing linked lists of dynamic data (e.g. mission data) that are updated during scenario execution.

Control for the initialization module is a simple set of calls by the Executive to three routines prior to simulation execution. The first of these routines (Initsim) performs all interfaces with input structures. This includes all interfaces with the INGRES data base and flat file input describing sensor converage. After this is completed, the Exec calls one of two routines (that perform specialized processing required to initialize the C^2 and Communications Modules. These routines use data loaded into the common blocks by the first initialization routine and its subroutines. There is no required call sequence between the second and third routines.

The reason that there are separate routines for the initialization of the C^2 and the Communications Modules is to accommodate the possible assignment of these functions to separate processors other than the CPU hosting the main simulation. Originally, this distribution of computational load was considered as a design solution that might be necessary to meet real-time processing requirements. As the design matured, it proved unneccessary to employ this approach in order to accomplish real-time execution. These initialization modules, however, were left separate in order to retain configuration flexibility.

If the user elects to distribute the functional modules, the common block interfaces to the C^2 and Communications Modules must be supported on the remote hosts. The capability to support the initialization of local routines in the Object Module was provided to support many of the common block synchronization requirements between the main and remote simulation hosts. These features of the ADISC^2 design will provide a backbone to support this distributed configuration if it is required.

Initialization Interfaces. The diagram presented as Figure 13, illustrates the interfaces from the initialization module to the other ADISC^2 modules. The interfaces from simulation initialization to the other run-time modules are three-way:

> 1) through common block initialization this module communicates the user-scripted scenario to all run-time modules;
>
> 2) through messages broadcast to remote nodes (including MMI); and
>
> 3) through the event stack, this module invokes user-prescripted events as per scenario definition and invokes periodic calls to simulation support routines.

The periodic simulation support routines that are invoked in this manner include object

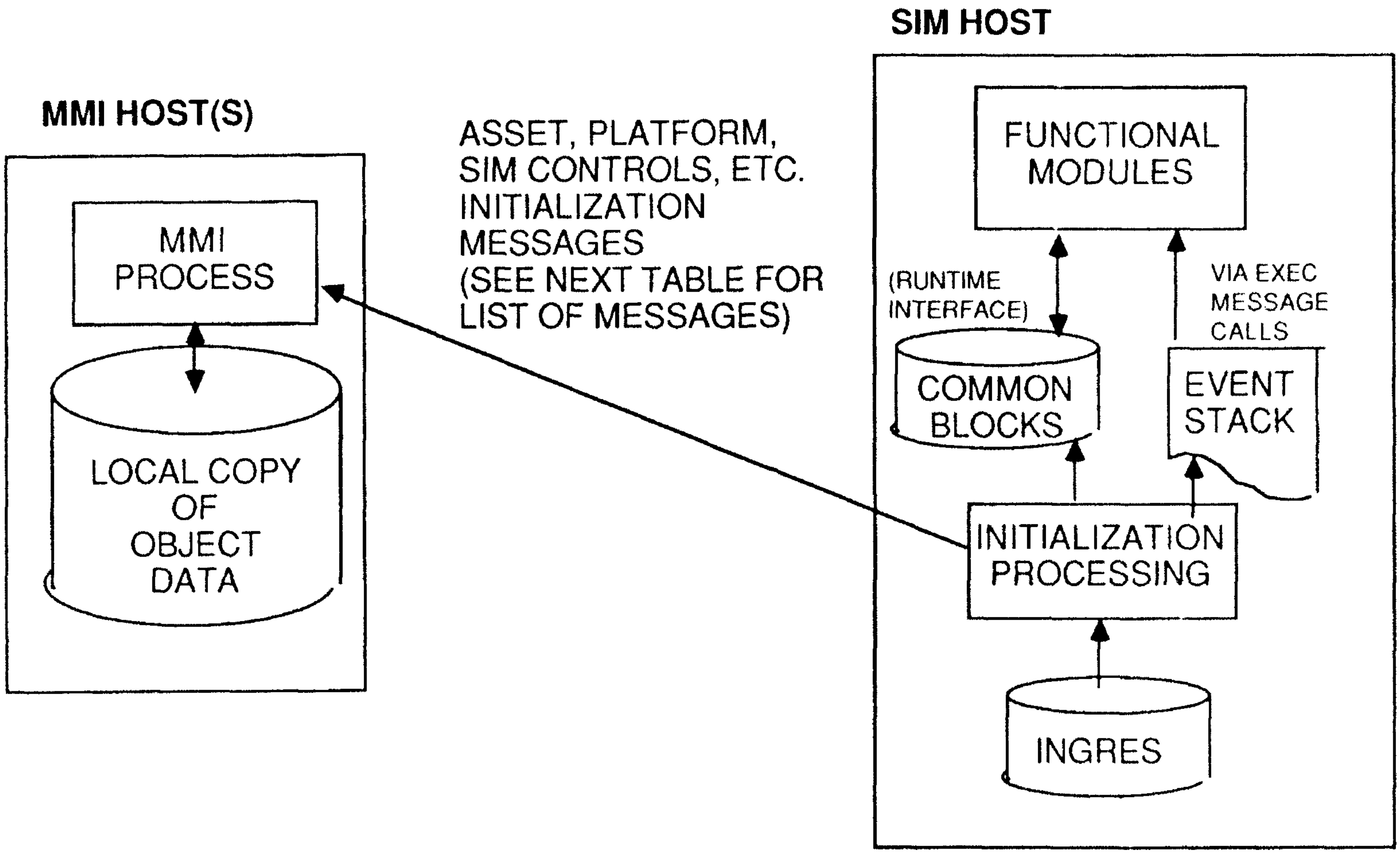

MMI HOST(S)
MMI PROCESS
LOCAL COPY OF OBJECT DATA
ASSET, PLATFORM, SIM CONTROLS, ETC. INITIALIZATION MESSAGES (SEE NEXT TABLE FOR LIST OF MESSAGES)
SIM HOST
FUNCTIONAL MODULES
(RUNTIME INTERFACE)
VIA EXEC MESSAGE CALLS
COMMON BLOCKS
EVENT STACK
INITIALIZATION PROCESSING
INGRES

FIGURE 13 - INITIALIZATION MODULE INTERFACES

MESSAGE NAME	DESCRIPTION	DESTINATION MODULE
Msgassetinit	Asset Definition	MMI, Object/Utilities
Msgplatforminit	Platform Definition	MMI, Object/Utilities
Msgtargetinit	Target Definition	MMI, Object/Utilities
Msgsensordefn	Sensor Performance Defn	MMI, Object/Utilities
Msgendinit	Initialization termination/ Scenario Definition	MMI, Object/Utilities
Msgairbaseinit	Airbase Definition	MMI, Object/Utilities
Msgacapupdate	Initial Auroral Cap Defn.	MMI
Msgchaffcu	Chaff determination	Environment
Msgbombeffects	Prescripted explosions	Environment
Msgnewwx	Prescripted weather	Environment
Msgwxupdate	Weather Movement Update Command	Environment
Msgassoclist	Associate Asset lists	MMI, Object/Utilities
Msgsensorinit	Sensor Site Definitions	MMI, FAA (Environment)
Msggroupinit	Group (Squadron) Definition	MMI, Object/Utilities
Msgchemeffects	Prescripted Bio/Chem Attacks	Environment
Msgcheckpoint	Check Point Instructions	Executive
Msgc2operate	C2 Operating Instructions	C2
Msgcommlinkinit	Communications Link Defns.	MMI
Msgengasset	Engagement Asset Defn.	MMI
Msgephemerides	Satellite Ephemerides	MMI
Msgsensorko	Sensor Detection Instructions	Sensor
Msgjmmrcmmnd	Prescripted Jamming Commands	Threat
Msglnchcms	Prescripted Cruise Missile Attacks	Threat

Figure 14 - Typical Messages Generated During Initialization

motion, sensor detections for each type of sensor, environment updates, and asset position reports to the MMI nodes.

The table, presented as Figure 14, lists the primary messages employed during Initialization. These represent the message interfaces to the other remote nodes and include the initialization messages output by routines in the Objects Module directory called at the completion of initialization. This is done because these messages are logically part of the initialization process. These messages are in addition to common block initialization performed by this module. The C2, environment, and communicaitons modules also send initialization messages to remote nodes. A complete list of the messages employed by the ADISC2 application is documented in the Interface Control Document. This list of messages is flexible and can readily be changed for alternate applications of the SSE, as demonstrated in a recent tactical application. The processing performed by these routines is described in the processing section of this module's documentation. Initialization module interfaces are output only.

ADISC2 Simulation Execution

Once a scenario has been selected from archive (either composed from a selection of parts/ elements, or generated from scratch) and initialization processes are complete, control is passed to the Simulation Executive for execution. Upon receiving run-control specifications, the simulation controller sub-function of the Simulation Executive formats the "execute simulation" message and that message is passed by the Executive's message-passing sub-function to all active nodes, with the routing determined by the configuration specifications produced in Pre-Simulation. Figure 8 (see page 9 (b)) presents a graphical representation of the functions involved during ADISC^2 run-time. Figure 15 is included to illustrate the flow of processing activities during ADISC^2 execution.

Prior to execution, the Executive resident at each node "checks in" with the Central Executive, known as "Exec Zero" (i.e., that Executive designated as being resident on the CPU that is the primary simulation host). At this point, the Executive starts the simulation clock. The Executive then checks the periodic event stack to determine whether there is an event to be executed. The periodic events are those regularly scheduled dynamics (e.g. sensor sweep-cycles occurring every "n" seconds) that produce simulated events (e.g. detection).

In addition, the Executive polls the ports it controls to determine if any command messages have been received during this clock step. If an event is scheduled that involves computation to be performed by one or more functional modules, the Executive routes messages that, in turn, call the appropriate subroutines. As outputs are generated by the subroutines, messages are constructed and formatted that are used to transmit parametric value changes and object status changes through the Executive services to the run-time database (which is configured much like a shared common in a FORTRAN program). The Executive time-tags each output message and inserts them, on a first-in-first-out (FIFO) basis, into the message queue.

It is important to note that real-time execution of a simulation may be impacted by the relatively slow execution of any of the called subroutines. This is because the simulation control is passed to the called routine. There are alternative methods for

FIGURE 15 - ADISC2 PROCESSING THREAD/SIMULATION

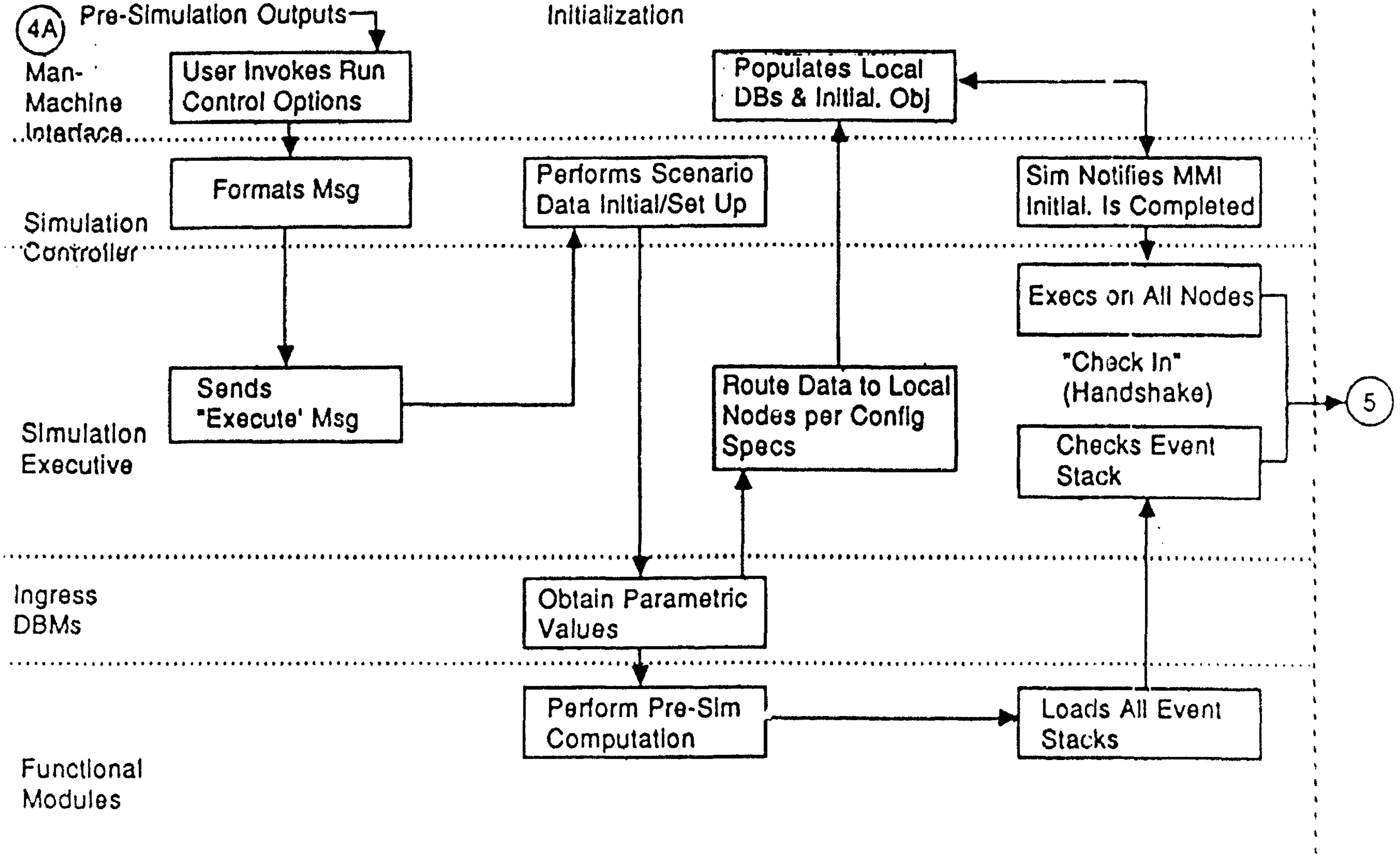
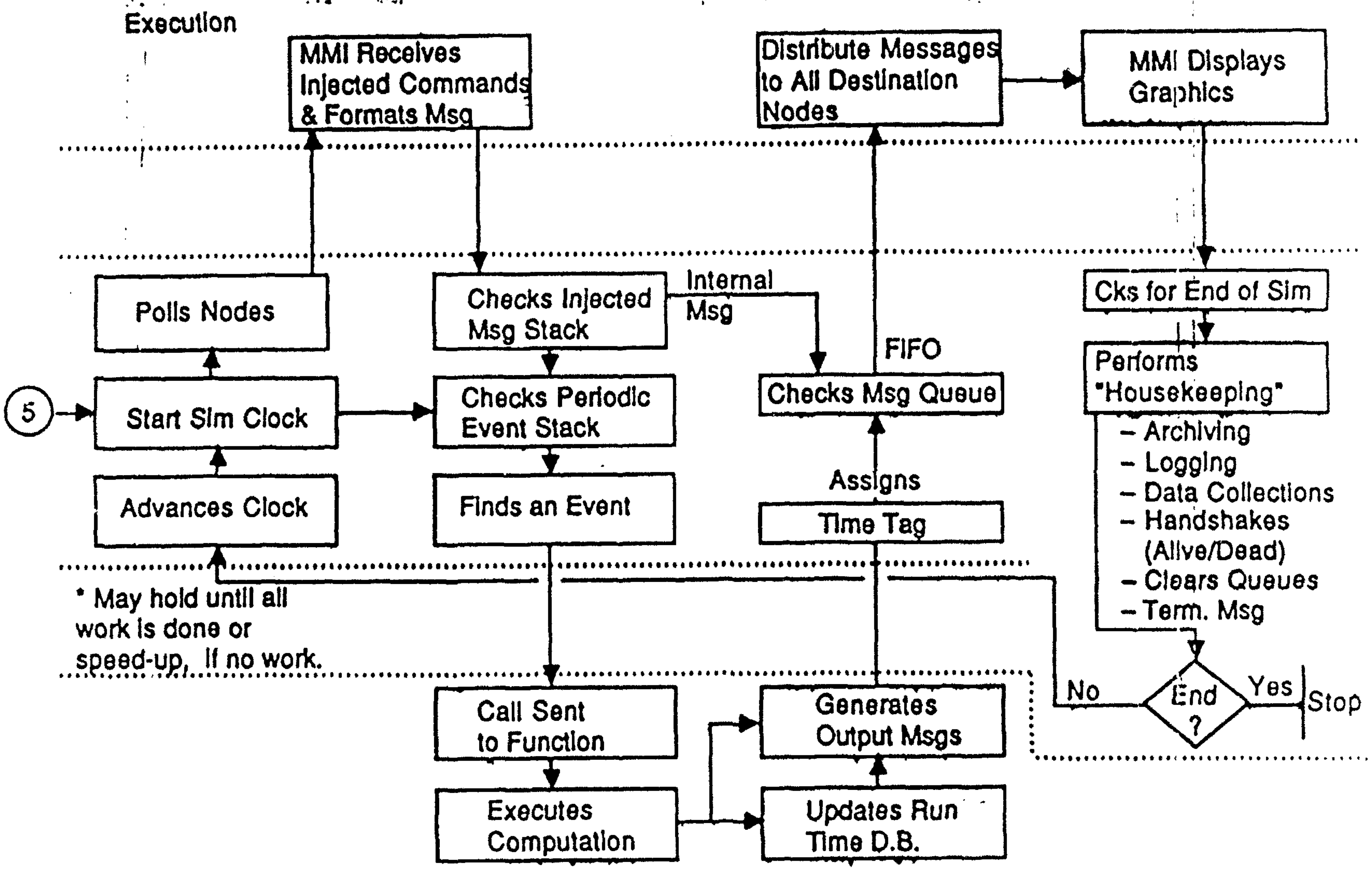


FIGURE 15 - ADISC2 PROCESSING THREAD/SIMULATION (CONT)



message control, timing and synchronization that are available to the user. However, the current design configuration is consistent with the provisions and specifications in the ADISC^2 SOW and maintains compatibility with the NTB p[illegible]

Possible Executive clock-control sub-function implementations would be adaptable to alternative methods for simulation execution. Time steps could be rigidly enforced and executed according to an inflexible schedule, regardless of whether all messages time-tagged for that interval are processed or not. The design currently employs an alternative method, in which the advance to the next time-step is delayed until all pending work and associated message processing has been completed by the Executive. This method of clock-control is compatible with most of the legacy software used in the construction of the air defense applications for ADISC^2 and guarantees consistency in simulation outcomes from one execution to another.

Slight variations in outcome could be introduced by using either of these alternatives. This led to the design decision to offer the user the capability to invoke any of these execution options and to mix-and-match clock-control and file-control features to tailor the SSE to the users' analytical requirements. Within the current SSE design, the user may elect either of the clock-advancing options in combination with queue-handling options of FIFO, last-in-first-out (LIFO) or priority handling according to user-defined data attributes. It should be noted that analysis of the ADI dynamics suggests that there are few real-world activities that would be sufficiently sensitive to time, in units of seconds or smaller divisions, that would perturb engagement outcomes significantly enough that the user should be concerned by various processing options in conducting C^2 evaluations. However, should such concerns arise, the SSE should be able to accommodate most requirements.

The Executive distributes messages to all destination nodes, including the display specifications that control animation of icons and changes to display graphics. The MMI creates the graphic display presentations using the COTS FIGARO capabilities incorporated into the design. At each time-step, the Executive checks for "end-of-simulation" message and performs various "housekeeping" functions such as log-in, data collection, configuration checks, network maintenance, and issuing of archive specifications. If the simulation is not at the end, the Executive advances the time clock and the above cycle is repeated until the simulation is completed.

ADISC^2 HARDWARE/SOFTWARE ARCHITECTURE

The design of the ADISC^2 is "open system" (ie., machine independent to the extent that the software requires a minimum amount or no modification to operate on most standard COTS hardware). However, this commitment to machine independence is no guarantee that the ADISC^2 software will perform equally in all hardware environments that are capable of supporting execution. In some cases this is obvious (ie., a graphic workstation capable of executing at a rate of 40 million-instructions-per-second (Mips) will provide faster responses and will support larger scenarios than a workstation capable of only 4 Mips). In other cases, the expected variance in performance is not so easily predicted (ie., there is a broad range of variance in the ability of various COTS hardware configurations to support X-Window or graphic standards without requiring special driver interfaces to invoke/employ graphics acceleration features of the workstation).

The primary advantage offered to the Government by an open system design is the reasonable assurance that the delivered software will be compatible with future COTS hardware releases. As graphic workstation technology has evolved rapidly in a competitive marketplace, the cost/performance curves for graphic workstations have trended steeply over the last few years in favor of higher speed and more computational and graphic capability for fewer dollars.

The ADISC^2 open system solution offers those considering acquisition of a hardware platform the option of selecting the highest performing workstation(s) at the lowest cost without a great concern over ADISC^2 software compatibility with a preferred vendor product. There are, however, certain preferred features and minimum specifications for graphic workstations and other elements of the hardware architecture that need to be considered to insure the optimum performance of the ADISC^2 software.

As of the date of this document, the ADISC^2 software is optimized to run on SUN 4 platforms. Silicon Graphics IRIS 4D/200-series machines offer better overall support for the ADISC^2 interactive graphics functions and are considered a better investment for the dollar than the SUN 4 workstations. However, the anticipated release of INGRES database software for the IRIS has not been accomplished by the vendor. Therefore, until the INGRES database software is available for the IRIS, there will be a need for a SUN server-node in the network configuration unless alternative, SQL-standard, relational DBMS products are used (ie., ORACLE, SY-BASE, etc.).
Not all SUN workstations are capable of supporting a fully functional ADISC^2. A platform capable of incorporating 32 Mbytes of RAM is recommended for all nodes to be used interactively with the simulation in a multi-node configuration. A workstation capable of supporting 128 Mbytes of RAM is recommended for the server-node that hosts both the simulation and the DBMS functions.

SUN offers a range of graphics options; the GX option is preferred. The requirements for shared memory may vary significantly with the size and complexity of the various scenarios that the ADISC^2 SSE is capable of supporting, but a minimum of a 669 Mbyte disk is recommended. Certain peripherals are useful such as a 1/4 inch (150 Mbyte) tape drive and Tektronix color plotter.

Target Computer System Specification. The ADISC^2 software can run in any of several network configurations, from one node to multiple nodes in a wide or local area network. The computer system for which the current version of the ADISC^2 software has been built and is targetted is the SUN Microsystems 4/200 family. The software has been successfuly ported to the SUN 4/400 RISC architecture series, the IRIS 4D series (with the considerations mentioned above) and the APOLLO 660 workstation. Networks may be configured with different vendor platforms acting as interactive nodes. The following details the system configuration and options of the development environment of the ADISC^2 software:

> **Batch Configuration;** Capable of execution of medium to large ADI scenarios (RADC "Mass Attack" scenario can be executed at about 3:1 real-time) in batch mode, without MMI graphics. This configuration offers fully functional support, including interactive graphics, for all Pre-simulation and Post-processing Analysis, but does not accommodate MMI interaction

during scenario execution.

- SUN 4/280 CXP (1 ea.; includes 9-track Tape Sub-system with 1600/6250 BPI)
- 64 Mbytes of RAM
- 19 " Color Monitor (1152x900)
- CG5/GP2 Color Buffer and Graphics Accelerator
- Mouse and Keyboard
- 130 Mbytes Process Swap Space

One-node Configuration; Also capable of execution of medium to large ADI scenarios, this configuration offers fully functional support, including interactive graphics, for all Pre-simulation and Post-processing Analysis, but accommodates only the White MMI node interaction during scenario execution. Here too, limitations exist with respect to execution of scenarios in real time with full communications, jamming and nuclear effects unde· this configuration.

- SUN 4/280 CXP (1 ea.) with 32 Mbytes RAM, 1 Gbyte Disk and GX Graphics option (includes 9-track tape sub-system with 1600/6250 BPI)
- SUN 4/260 CXP (1 ea.) with 32 Mbytes RAM and GX Graphics option
- 1/4 inch (150 Mbyte) Tape Sub-system (1 ea.)
- 8mm (Exabyte) 2.3 Gbyte Tape Sub-system (1 ea.)
- Ethernet (1 ea.) with connection for each workstation

Two-node Configuration; Provides full MMI interactive simulation support for two participants with simultaneous views of either White or Red and Blue or White or Blue and Blue. This configuration can support execution of all current ADI scenarios in real time if the system is dedicated to this purpose exclusively.

- SUN 4/280 CXP (1 ea.) with 32 Mbytes RAM, 1 Gbyte Disk and GX Graphics option (includes 9-track tape sub-system with 1600/6250 BPI)
- SUN 4/260 CXP (2 ea.) with 32 Mbytes RAM and GX Graphics option
- 1/4 inch (150 Mbyte) Tape Sub-system (1 ea.)
- 8mm (Exabyte) 2.3 Gbyte Tape Sub-system (1 ea.)
- Ethernet (1 ea.) with connection for each workstation

Three-node Configuration; Offers full interactive MMI support to three simultaneous participants (generally White/Ground truth, Red perception and Blue perception or White, Blue and Blue). This configuration can support three simultaneous views of the scenario by three interactive participants. All ADI scenarios can be executed at real time or faster. Some interactive loadings will cause slower execution with jamming and full-up communications loadings in large scenarios.

- SUN 4/280 CXP (1 ea.) with 32 Mbytes RAM, 1.5 Gbyte Disk (includes 9-track tape sub-system with 1600/6250 BPI)

- SUN 4/260 CXP (3 ea.) with 32 Mbytes RAM and GX Graphics option
- 1/4 inch (150 Mbyte) Tape Sub-system (1 ea.)
- 8mm (Exabyte) 2.3 Gbyte Tape Sub-system (1 ea.)
- Ethernet (1 ea.) with connection for each workstation

Additional Blue nodes, up to sixteen, may be added.

Simulation Support Environment Software. The design of ADISC^2 was achieved with the maximum use of COTS software, as per the contract requirements. The following is a listing of the software licenses required for the ADISC^2 application:

PRODUCT	CURRENT VERSION	CURRENT RELEASE *
SUN OPERATING SYSTEM	4.0.3	4.1
(With Patches for DBX, Shared Memory, Bus Panic, etc.)		
INGRES	5.0	6.0

PRODUCT	CURRENT VERSION	CURRENT RELEASE *
TEMPLATE/BLOX	6.0/3.0	----
FIGARO	1.2	2.0
SUN FORTRAN	1.2.1	1.3
NAG LIBRARY	13	14
SUNLINK TE-100	6.0	----
or		
VT-100 WINDOW MGR		

* Plans are being made to upgrade the ADISC^2 tested baseline to the current release under on-going O&M and configuration management tasks and should be completed by early 1991.

LESSONS LEARNED

Over the course of the development effort, there were significant technical challenges to be addressed and numerous lessons were learned by both the Martin Marietta and the Government staffs.

During the course of the program, a considerable number of problems with the various releases of SUN Microsystems Operating System (O.S.) software were encountered. Some of these are known and fairly well documented, some are not widely understood by much of the user community. Some of these problems have their origins in the area of mis-communication or lack of communication between a highly specialized, over-worked and fragmented technical support group at SUN and their customers. Some problems are characteristic of a rapidly changing hardware environment where software releases appear to be made prematurely in order to keep pace with the latest hardware advance in an intensely competitive marketplace.

All of the known problems with the SUN systems and software have been addressed and corrected at the time of this writing. Generally, SUN staff members have been helpful and supportive in the process of correcting deficiencies and defects; once one is able to find and get the attention of the knowledgeable people, that is. This section is provided as an attempt to increase the awareness of those who may acquire ADISC^2 software and attempt to host it on a SUN system. Hopefully this section may prevent others from undergoing the difficulties that have been experienced in using and debugging SUN COTS software.

Problem: Network File System (NFS) Page Buffer Overflow. This problem was first noted after installation of the O.S. release 4.0.3, in August, 1989. Simulation executions generating relatively large files would abort when the system utilization reached levels of 80% or more. The system would only produce brief and cryptic diagnostic messages, "msgmap: rmap ovflo. lost [2357,2361]", for which there was no reference in the SUN documentation.

After several unsuccessful attempts to obtain technical assistance from the SUN organization, RADC was notified of the problem in the Program Monthly Status report for October, 1989. Initially this problem was only experienced in larger simulations than were required for the successful completion of the contract. However, with the larger scenarios ultimately provided by the Government, this problem occurred frequently and caused many simulation execution failures. With the help of a SUN technical manager, a parametric change was made to the O.S. configuration specifications that eliminated this problem.

Problem: Network File System (NFS) File Corruption. It was noted subsequent with the installation of the 4.0.3 O.S. software, that unexplainable changes were being introduced into the ADISC^2 source-code listings after they had been tested and placed under configuration management control. Lines of source code would disappear, nulls were being introduced at various locations, extraneous control characters were being inserted, blank lines and spaces would randomly appear, and lines were being duplicated randomly between simulation executions. These variations would be noted in code that had been successfully compiled and executed, only to fail when re-linked for subsequent executions.

When a technically knowledgeable SUN staff member was found with an explanation for this problem, several test sequences of the Preliminary Acceptance Test had been failed. It so happened that SUN Microsystems had developed several "Patches" for their O.S. software that addressed this and other NFS problems about four months earlier but had been releasing these fixes only to those sites reporting problems to those technical groups responsible for the repair problem. Some of the technical staff at the software "Hotline" were apparently unaware of the "Patches" and did not provide the appropriate responses over a period of several months of frequent complaints. Even though maintenance agreements existed, including "Hotline" services, SUN insisted that the only way to be assured of receiving the proper protection from COTS bugs was for the Contractor to pay additional fees for a dedicated SUN staff member who would serve as a dedicated point of contact for problem resolution at the development site.

Problem: Network File System (NFS) Write Errors, Segmentation Errors & Bus Errors. Over the course of the development period, numerous unexplainable errors were noted without sufficient documentation or diagnostic support provided by the SUN system to expediently address the problem. Only limited diagnostic support is provided by the SUN environment with no traceback to the fault location.

This difficulty was frequently compounded and made more aggravating when attempts were made to communicate problems to the SUN technical staff. One of the first questions asked was, "Where did the execution terminate?" as well as equally unanswerable questions due to the limited capabilities of the DBX Tool for Fortran traceback. It was interesting to experience a relationship with "Hotline" staff offering assistance predicated on information that, if available, would have eliminated the need for assistance in the first place. Many times, the only recommendation provided was, "Do a core dump". The size of such a dump consumes a very large amount of available memory (many times requiring memory space not available at all) and requires considerable time to create the output and then to reload to memory under the DBX Tool. In addition, the process locks out other activities in the development environment. Under a fixed-price effort, such inefficiency is unacceptable.

Problem: Command Tool Window Overflow. On numerous occasions, an overflow of the command tool window has been experienced. Frequently, the window would mysteriously disappear without any explaination. This is caused when the amount of information developed by a simulation during execution exceeds the system limit on the number of bytes that may be stored in a window. When this limit is exceeded, the system converts to a mode of non-scrolling shell tool window and much valuable information is lost. Solutions include redirecting the data to an alternative disk file rather than the screen and then do a "tail" to produce displays of the information or periodically clearing the data from the screen during the course of the execution.

CONCLUSIONS

The successful development of the ADISC2 Simulation Support Environment has validated the design hypothesis that:

1. It is possible to achieve real-time or faster executions of an interactive, distributed simulation, incorporating the numbers of objects involved in a realistic representation of the NAADE, and operating at a level of fidelity and granularity required to support C^2 development and evaluation;

2. It is also possible to integrate widely disparate and independently developed legacy simulation software within such a distributed simulation environment and that such integration can be accomplished at a level-of-effort and cost below that required by comparable new code development activities.

3. Re-use of modular legacy software is possible and practical and may be attained by using message-passing paradigms to overcome software machine-dependency and language dependency without requiring costly and time-consuming translation of legacy code to an Ada language standard. It is also demonstrated with IR&D associated with this effort that certain software functions related to graphics and interfaces to operating system software can be more efficiently accomplished by integration of legacy using such High-Order Languages (HOL) as C or C++, than may be possible with Ada;

4. It is possible to exploit object-oriented paradigms to accomplish a high level of flexibility in simulation object definition and that it is possible to employ these objects in a wide range of simulation applications at varying levels of fidelity as required to satisfy diverse analytical needs;

5. Commercial Off-The-Shelf (COTS) software and hardware exists that are capable of meeting run-time requirements for man-in-the-loop, C2 evaluation;

6. Various independent COTS software products can be economically integrated and interfaced with various simulation software applications and database management systems to meet the requirements for rapid prototyping; and

7. Compliance with existing software and network management standards does allow a relatively high degree of machine independence and transportability without significantly compromising performance requirements for C^2 simulation.

It has been established, through the development effort and the subsequent review of the ADISC^2 by many government officials and technical staff, that there are numerous potential applicatons for the ADISC^2 SSE technology. In addition to the use of ADISC^2 as a source/sink of threat and sensor data for prototyping future SOCC C2 operations, there are a number of Government organizations that either have installed the ADISC^2 software for other uses or have indicated an intent to transfer this technology:

ORGANIZATION	POTENTIAL APPLICATION
- ESD/XRT	Support ADI Architecture Planning & Acquistion
-NORAD/J-5	ADI Architecture Evaluation
-AIR STAFF/AQSD	Support Evaluation of Wide Area Surveillance Systems
-AIR STAFF/XN	Theater-Level, Air Operations Planning & Evaluation
-ARMY SDC	Army Air Defense Simulation
-SPACE COMMAND	Wide Area Surveillance Applications
-FTD/TQS	Threat Analysis
-NRL	Various Classified Applications
-JSTEP SAG	Adaptive Strategic Planning & Evaluation
-ARMY COMBINED ARMS CENTER	Ground Force (Coprs/Division) Simulation

It has been concluded from the technical achievements of the successful Phase I demonstration and Phase II implementation and testing that the current design of the ADISC^2 can support the following user-generated potential applications for air defense C^2 development:

- Determination of critical time parameters for C^2 decision-making;

- Identification and definition of C^2 primitives and the building of composite C^2 functions by mixing-and-matching these primitives;

- Assessment of performance and impact of alternative C^2 functional composites;

- Determination of critical inter-relationships among C^2 decision-making primitives;

- Identification of opportunities for automation of C^2 functions and the ability to empirically evaluate the impact of automation;

- Evaluation of alternative strategies on C^2 operations; and

- Definition of required data content for C^2 operations.

RECOMMENDATIONS

Automated Intelligent Archive/Configuration Management. With the flexibility and modularity provided by the ADISC^2 design, there is an associated level of complexity involved with the maintenance of archived scenario elements. Scenarios may be built from a nearly infinite number of combinations of threats (varying the number and 'ype of threat objects, routes, load-outs, weapon/target pairings, initial deployment schemes, etc.), alternate defense architectures and C^2 concepts, alternate sets of run-control parameters, various user-injected changes to pre-scripted scenarios, and numerous excursions from performance baselines achieved by selection of alternative parametric values. The ability to manage the likely geometric growth of archived scenario elements and the ability to efficiently mix-and-match scenario primitives and to properly apply these composites to specific problems will *require computer-aided solutions.*

The functions of archive maintenance and data management represent a most likely opportunity for incorporation of Artificial Intelligence (AI) technologies into future versions of ADISC^2. Since there are so many possible scenario baselines, numerical variations to scenarios, and so many combinations of parametric inputs that can be developed with the implementation of ADISC^2, an Intelligent Database Manager/User Interface will be an extremely important future enhancement of ADISC^2.

The development of an intelligent archive/data manager will require identification of significant key attributes associated with scenario elements and development of a rule-base for mapping simulation applications to these keys. The construction of such an intelligent interface to scenario elements will allow ADISC^2 users to present desired simulation outcome specifications to the system and invoke rule-based scenario options capable of satisfying stated purpose. Such a capability will also allow future users to request a specific set of scenario primitives and the AI program will select that scenario, or some subset of scenarios that most closely meet the users' specifications. This capability will greatly reduce redundant growth of scenario files and greatly reduce the need to create new scenarios when modification of an archive would achieve the users objective. An artificial intelligence capability for archive maintenance should be incorporated at the earliest possible date.

Enhanced Sensor Representation. The scope of the current ADISC^2 contract and budget as well as the current state of development of some of the technologies such as

Space-Based Radar (SBR) and OTH-B (particularly the lack of definition of operational protocols and procedures) resulted in limited representation of these elements of the NAADE in the design. These simulations should be enhanced at the earliest opportunity.

In the case of OTH-B, the simulation enhancement recommended would include the development of man-machine interface capabilities to allow the operator to place the outer barriers and the three discretionary views within the twenty-eight second sweep cycles to optimize track detection and track correlation. This would provide a significant improvement in the degree of realism and fidelity of simulated detections involved with C^2 development applications. In the case of SBR, similar development is recommended in providing mechanisms for positive man-in-the-loop control of directed positioning of the sensor to a specified scan area and further development of optional representations of higher fidelity phenomenology associated with sensor operation and detection events.

Demonstration of Automated Decision Aid Evaluation Capability. The ADISC^2 design is compatible with most Artificial Intelligence software applications due to its object-oriented approach. Experiments to evaluate the effectiveness of C^2 decision aids and option generation software may be accomplished with relatively small investment in time and money. As an example, the Conus Air Defense Resource Allocation Advisor (CADRAA) decision aid for air defense assignment against threat tracks could be evaluated for timeliness and impact on simulation outcome by hosting the ADISC^2 Executive software on the same host as CADRAA. Messages describing the situation represented by the simulation would be passed as input to the decision aid program via Exec-to-Exec transfer Command and messages generated by application of the CADRAA rule-base would be passed to the simulation. Since the simulation is operating in real time, the timeliness of the response by the decision aid and the impact on simulation outcome may be quantified and evaluated. Such experiments can be conducted in the near future and are recommended as an extention to the current contract.

Improved Blue Node Capabilities. The scope of the current contract and budget limits the extent of the development of the defense planning interface for ADISC^2. The commitment made within this effort was to provide a user-friendly, menu-driven and interactive graphics interface allowing rapid deployment and re-deployment of engagement assets. This interface allows Blue Node operators to address the most likely threat approaches, to perform situation assessment relative to perceived threat movement and to deploy defense elements to more advantageous positions (eg., assign interceptors to CAP) in an attempt to achieve a more favorable outcome. In order to exploit the potential of ADISC^2 to support defense architecture concept development and evaluation as well as the C^2 development function that is its primary function, considerable modification would be in order. Such enhancement would include development of new man-machine interfaces tailored to the processes of defense architecture concept development. In addition, there should be additional interfaces to the analytical software capabilities recommended above.

These interfaces would allow the Blue Node operator to invoke statistical routines during simulation interrupt. With this capability, ADISC^2 users could analyze parametric data from partially completed scenarios, graphically display aggregated data, establish trends and statistical correlations and then checkpoint to a prior time step or simulation event in order to replay with interactive modifications. This capability could also provide ADISC^2 users with the option to continue from the point of interrupt with subsequent interactive command and control actions exercised with the benefit of information derived from statistical analyses. By performing controlled excursions from simulation baselines, with the support of integrated statistical analyses, it will be possible to optimize performance outcomes of alternative defense architectures and operational concepts.

Enhanced Communications Capabilities. A number of significant opportunities exist for enhancement of the Communications Module of the ADISC^2. The current design allows the rapid configuration of alternative communications networks for support of air defense operations. Current capabilities allow for alternative definition of node connectivity and specification of a wide range of link-link performance parametrics, and the ability to analyze message throughput and propagation characteristics. The impact of various levels of message loading on the probability of correct message receipt and the associated impact of message loads on engagement outcomes may be assessed. The ADISC^2 design also allows assessment of the impact of node and link losses on various communication systems and networks. Proposed future enhancements of the Communication Module would include the design and development of algorithms to find alternative network paths among nodes to direct message traffic in stressed environments. In developing this capability, it may be feasible to incorporate legacy software from existing communications systems in order to represent connectivity in a realistic fashion.

At present the communications module creates a virtual "time out " when the fifteen minute update is reached. Several schemes to improve this, including maintaining a linked list of the active communications links for the update, have been discussed. The savings in time could be substantial. Another area of time savings would be to develop a geographic database of "nearby" sites with similar communications equipment. In this case, nearby will also include sites such as satellites and HF sites which can see the originating site even though at long range or over the horizon.

A high priority in the area of improved fidelity of the communications module is the enhancement of the telephone system representation. The current telephone system modeled is a fully-interconnected network with any node completely interconnected to all other nodes. This is a reasonable first-order approximation. For networks and operating scenarios which are highly dependent on telephone communications for data transfer, this approximation gets progressively worse as more damage to a real system is expected. This situation certainly occurs in air defense scenarios in which major cities (where most telephone switching gear is located) are hit. The recommended upgrade is to build an underlying telephone switching model based upon existing public switched network models.

In the area of the radio-frequency and laser propagation models, a reconstruction of the most often used routines and data structures of the STRATC^{2}AM model is appropriate. At present, this model is bulky and unwieldy, having literally hundreds of common blocks and multiple redefinition of the same variables. If the main routines could be reworked, along with a restructure of the common blocks and an elimination of much of this redundancy, the code would be much more maintainable and one would suspect, faster. In addition, the current STRATC^{2}AM laser models consider only a single case.

The last recommendation in the area of the STRATC^{2}AM code is replacement of the nuclear effects code, which is an older DNA model called WESCOM, with a code that is currently supported by DNA (support for WESCOM was dropped several years ago). A code such as PRPSIM, which is available and supported would give a more credible nuclear effects representation.

Another enhancement to the communications code, which would be of particular use to the communications analyst, is to extend the communications man-machine interface. The present interface is limited to observing events and status, and tells the operator little about the operation of the links or equipment at the nodes. Adding capabilities found in commercial packages such as BOSS or BONES significantly increases the usefulness of such a tool for the communications segment of the C^3 community.

Another recommended enhancement to the Communications Module is the incorporation of automatic communications generation for various message events to allow saturation of selected links by invoking statistical distributions for message loading (eg., uniform, normal, beta, gamma distributions). Typing this enhancement to the communications MMI upgrade gives the operator the opportunity to inject traffic and immediately see the results of traffic loading on the operation of the entire command and control system.

Expanded C^2 Representation. The current C^2 module is designed primarily to support relatively limited simulation periods (2 days or less) and to control assets in response to perceived threat activities prior to and during hostilities. It has very limited planning functions, primarily invoking user-predefined operations plan. This has resulted in the following limitations that could be expanded in the future:

1) There are currently no provisions for reconstitution between attacks.

2) There are obvious provisions to hold assets in reserve in the expectation of a multiwave attack.

The structure of the C^2 module lends itself to expansion to incorporate additional missions to the strategic air defense mission. Possible future expansions based upon this module and the Red Mission Planner include incorporating tactical C^2 missions such as Air Interdiction (AI & BAI), Close Air Support (CAS), and Offensive Counter Air (OCA). Several limited scenarios have been successfully executed using the C^2 module during ADISC^2 check-out, and the results indicate that with minor modifications to reflect operational limitations, ADISC^2 and this C^2 module are directly applicable to the evaluation of Limited Intensity Conflict (LIC).

Currently the C^2 module allows the user to specify whether some stimuli will be responded to and under what conditions a stimulus should invoke a response. For example, the user can specify that the C^2 module should evaluate the utility activating a CAP at a given location based upon the perceived threat activity (as determined by tracks or intelligence reports) within a specified radius of that locale, but in general the user can not specify compound conditions to be tested for invoking a response unless these compound conditions are precoded. Based upon our current experience, an interactive interface offering support for users tc define stimulus-response behavior rules for automated simulation execution would be both feasible and very beneficial. This interface would allow users to interactively define tactics and strategies, Rules of Operation and Rules of Engagement. This concept, as illustrated in Figure 16, allows the user to specify the logic that the C^2 module is to employ in its behavior.

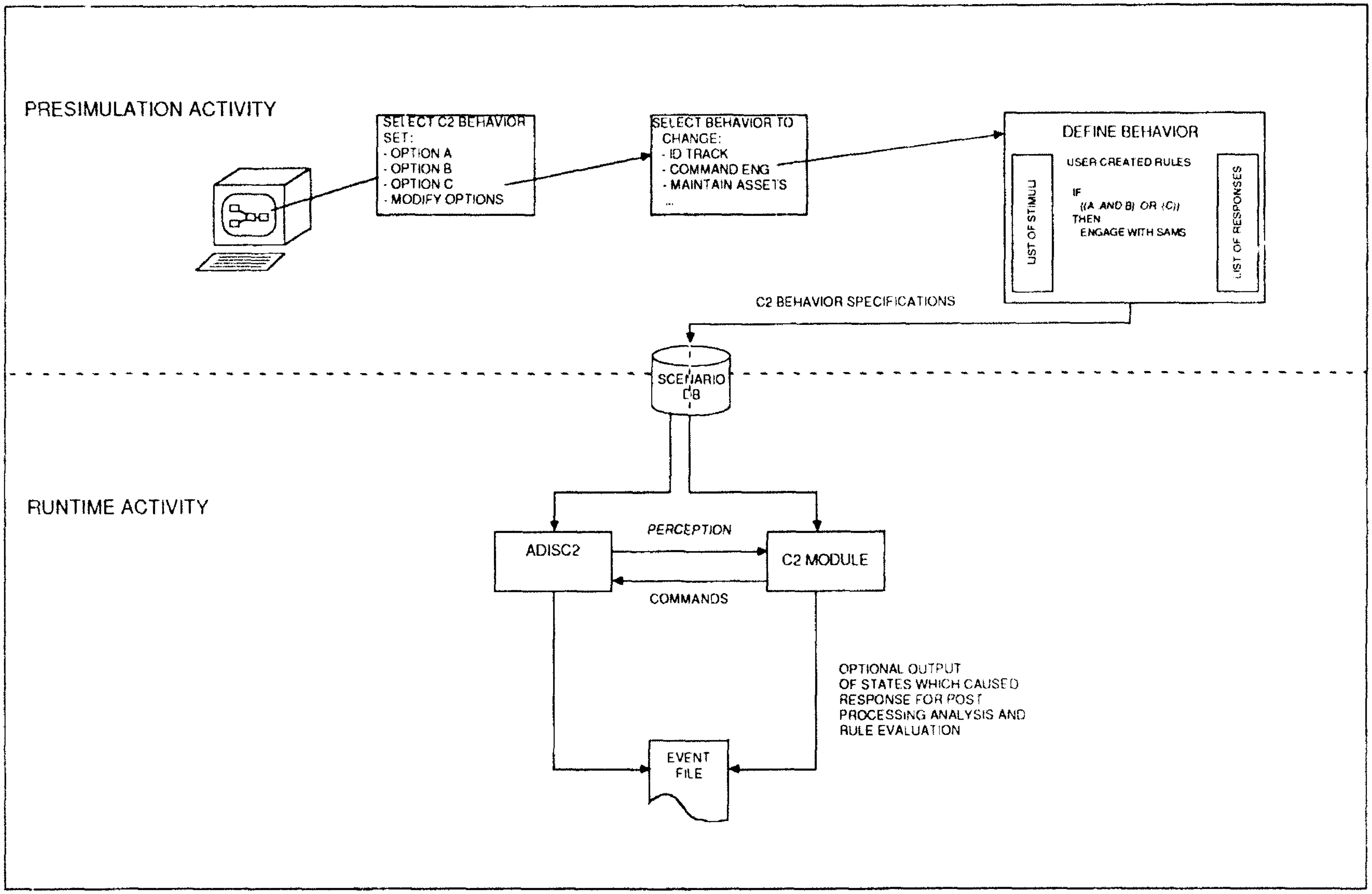


FIGURE 16 - C2 DEFINITION FOR ANALYST DEFINABLE STIMULUS/RESPONSE BEHAVIOR MODEL

REFERENCES

1. Rome Air Development Center, "Statement of Work for Air Defense Initiative Simulation for Command & Control Development" (ADISC^2); PR No. B-7-3134 14 May, 1987.

2. Skolnik, Merrill I., Radar Handbook; Mcgraw-Hill Book Company, New York, N.Y., 1970

3. Skoinik, Merrill I., Radar Design Principles; Mcgraw-Hill Book Company, New York, N.Y., 1960

ACRONYMS

AAM	Air-to-Air Missile
ADI	Air.Defense Initiative
ADISC^2	Air Defense Initiative Simulation for Command and Control
AF/S&A	Air Force Study and Analysis (Organization)
AI	Artificial Intelligence
ASW	Anti-submarine Warfare
CADRAA	Conus Air Defense Resource Allocation Advisor
CAP	Combat Air Patrol
COTS	Commercial Off-the-Shelf
CPU	Central Processing Unit
C^2	Command and Control
C^{2}TL	Command and Control Technology Laboratory
FSD	Full Scale Development
GFI	Government Furnished Information
JSS	Joint Surveillance System
MINI-MUF	OTH-B Model (NORAD)
MMI	Man Machine Interface
ML	Mobile Laboratory
NAADE	North American Air Defense Environment
NDBMS	NORAD Data Base Management System
NOSC	Naval Oceans Systems Center
NTB	National Test Bed
NWS	North Warning System
OTH-B	Over the Horizon-Backscatter
OTS	Off-The-Shelf
QUEL	Query Language
RADC	Rome Air Development Center
RESA	Research Evaluation and Systems Analysis
ROCC	Regional Operations Control Center
SAM	Surface-to-Air Missile
SBR	Space Based Radar
SDI	Strategic Defense Initiative
SOCC	Sector Operations Control Center
SOSUS	Sound Ocean Surveillance System
SOW	Statement of Work
SQL	Structured Query Language
SSE	Simulation Support Environment
SURTASS	Surveillance Towed Array Sensor System
UIMS	User Interface Management System

Appendix

The ADISC^2 is supported by a high-resolution graphics package that. provides the level of detail necessary to perform simulations of varying degrees of fidelity in real-time executions. Samples of these graphic images are included in this appendix.

Appendix Graphic Image Samples.

Figure Page

A1. Sensor Coverages. A-1

Illustrates sensor coverage areas for ground based sensors (Long Range Radars and OTH-B), ship based radar sensors, sub-surface sensors (SOSUS), airborne surveillance radars (AWACS), and space based radars (SBRs).

A2. Platter. A-2

This platter projection places the North Pole at the center of a two dimensional disk with the South Pole displayed 360 degrees around the perimeter. Satellites and constellations of all types (Communications, Radar, Navigation, Weather, Intelligence) may be added and displayed quickly.

A3. Targets. A-3

Hundreds of strategic targets in North America divided into eight categories (Military, Population Centers, Industrial Centers, Strategic Offense, Strategic Defense, C2, Communications, and National Command Authorities) may be displayed and utilized for strategic planning.

A4. Red Mission Planner. A-4

The Red Mission Planner allows the user to quickly plan a comprehensive strategic air attack on North America. In addition to designating the bomber attack forces, the air routes and weapon release points, the user may select the attack sectors, prioritize the targets, and assign the types and numbers of weapons (gravity bombs, cruise missiles) to be launched. As shown in the window inset, the cruise missile great-circle routes may also be displayed and examined.

A5. Attack Bomber Routes. A-5

The great-circle routes for designated attacking bomber forces are illustrated. Connectivity of airbases to staging areas, to weapon launch points, and to recovery bases can be defined or modified by the user.

A6. Air Defense Picture. A-6

The current air defense situation for both the attacking and defending forces may be viewed by the operator utilizing the White Node provided by ADISC2. As illustrated, the TRUE INFORMATION menu selection results in the display of the opposing airborne forces as well as nuclear detonation (NUDET) locations.

A7. Communication Links. A-7

Communication links employed in the scenario are displayed in this communications overlay. Many types of communications (ship-to-shore. air-to-air. ground-to-ground and other variations) may be illustrated.

A8. Air Engagement. A-8

Interceptor forces may be controlled by the operator. A wide variety of engagement controls, commands, and information may be selected directly from the menus. Other critical data (speed, altitude, location, fuel state, weapons status) concerning the status of the engagement forces are also displayed and may be utilized by the operators.

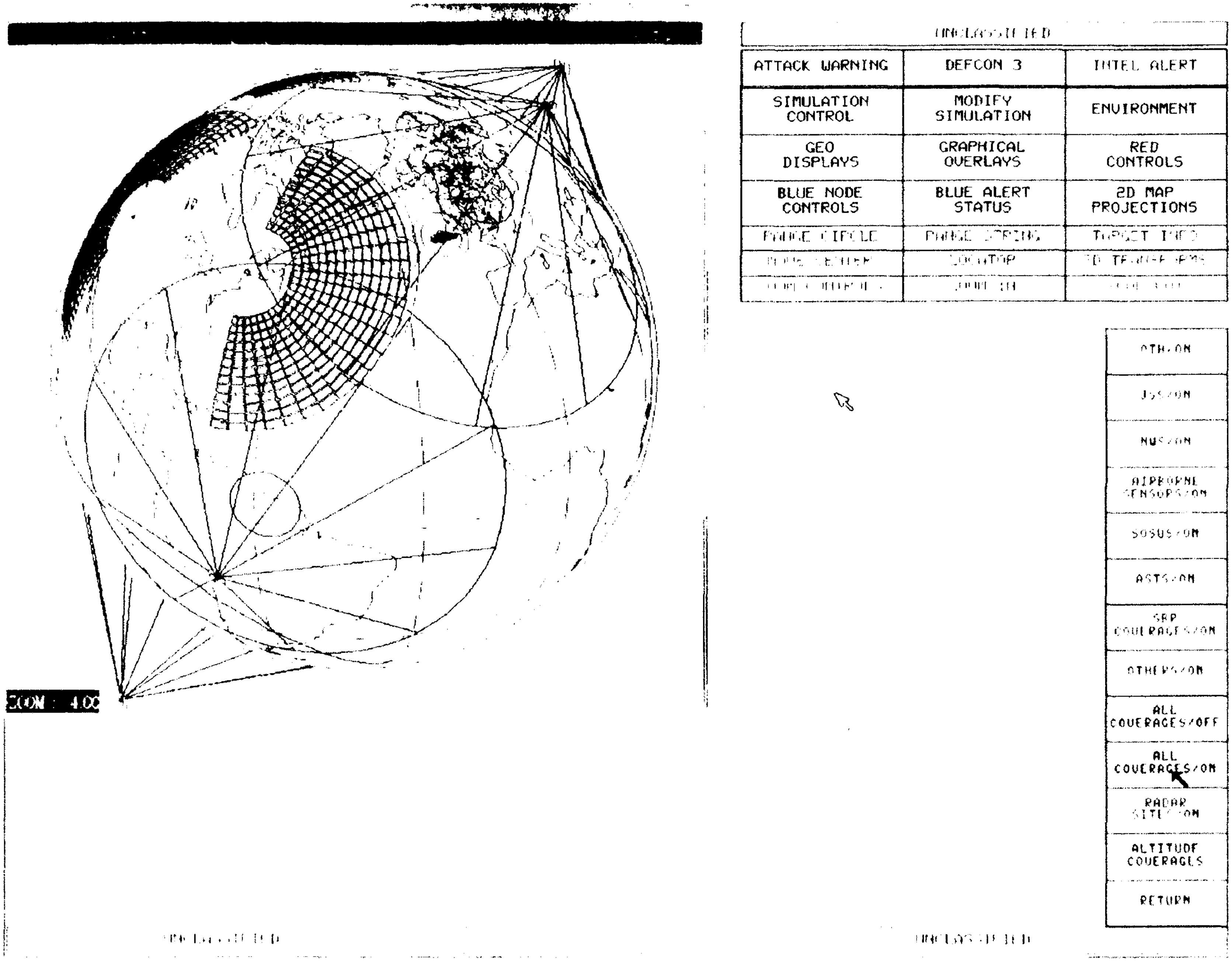
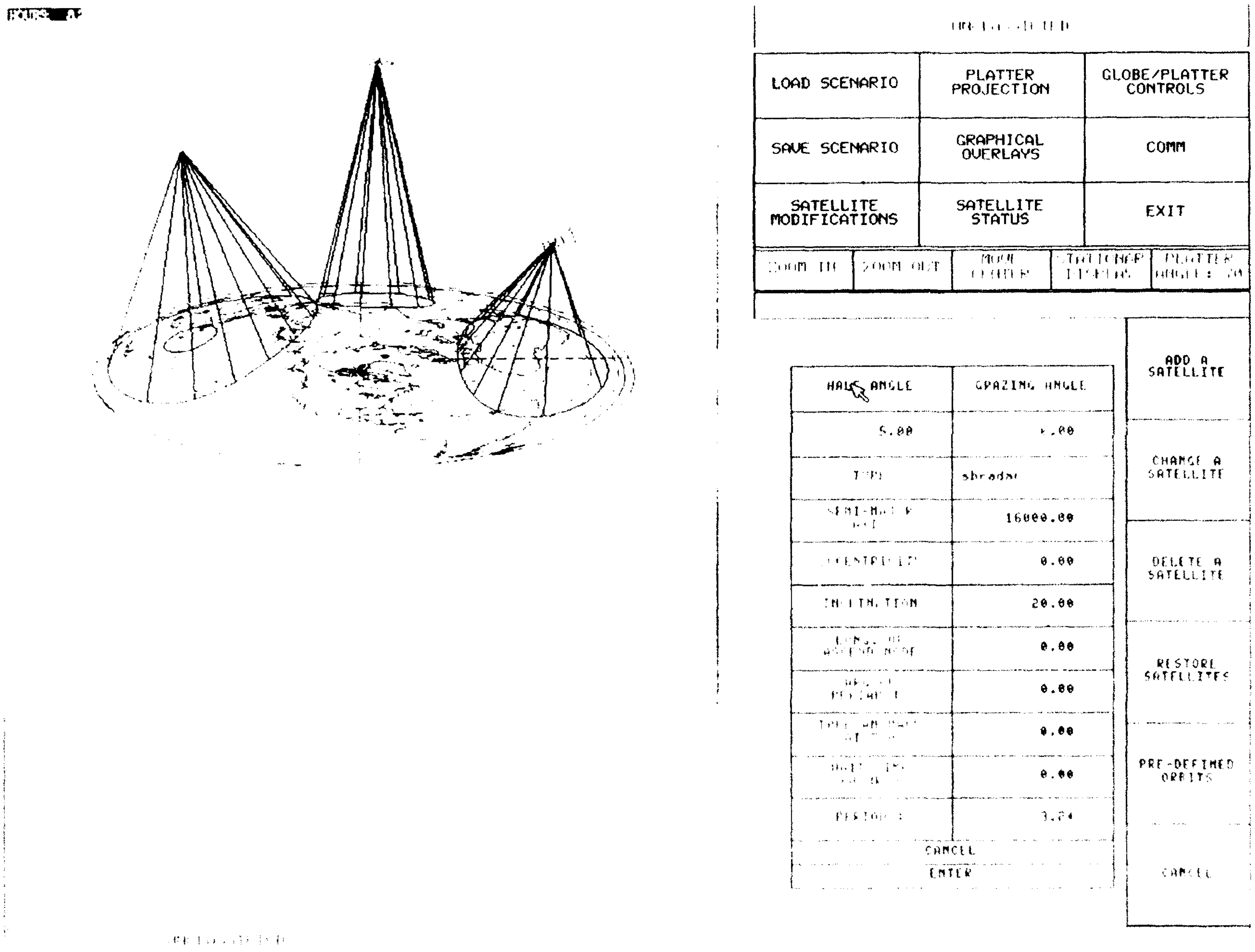

UNCLASSIFIED
ATTACK WARNING
DEFCON 3
INTEL ALERT
SIMULATION CONTROL
MODIFY SIMULATION
ENVIRONMENT
GEO DISPLAYS
GRAPHICAL OVERLAYS
RED CONTROLS
BLUE NODE CONTROLS
BLUE ALERT STATUS
2D MAP PROJECTIONS
RANGE CIRCLE
RANGE STRING
TARGET INFO
LOCATOR
3D TRANSFORMS
ZOOM IN
OTH/ON
JSS/ON
NUS/ON
AIRBORNE SENSORS/ON
SOSUS/ON
ASTS/ON
SRP COVERAGES/ON
OTHERS/ON
ALL COVERAGES/OFF
ALL COVERAGES/ON
RADAR SITES/ON
ALTITUDE COVERAGES
RETURN
ZOOM : 4.00
UNCLASSIFIED
UNCLASSIFIED

Fig A1



HALF ANGLE	GRAZING ANGLE
5.00	[illegible].00
TYPE	shradar
SEMI-MAJOR AXIS	16000.00
ECCENTRICITY	0.00
INCLINATION	20.00
LONG. OF ASCEND NODE	0.00
ARG. OF PERIGEE	0.00
[illegible]	0.00
[illegible]	0.00
PERIOD :	3.24

CANCEL

ENTER

Fig A2

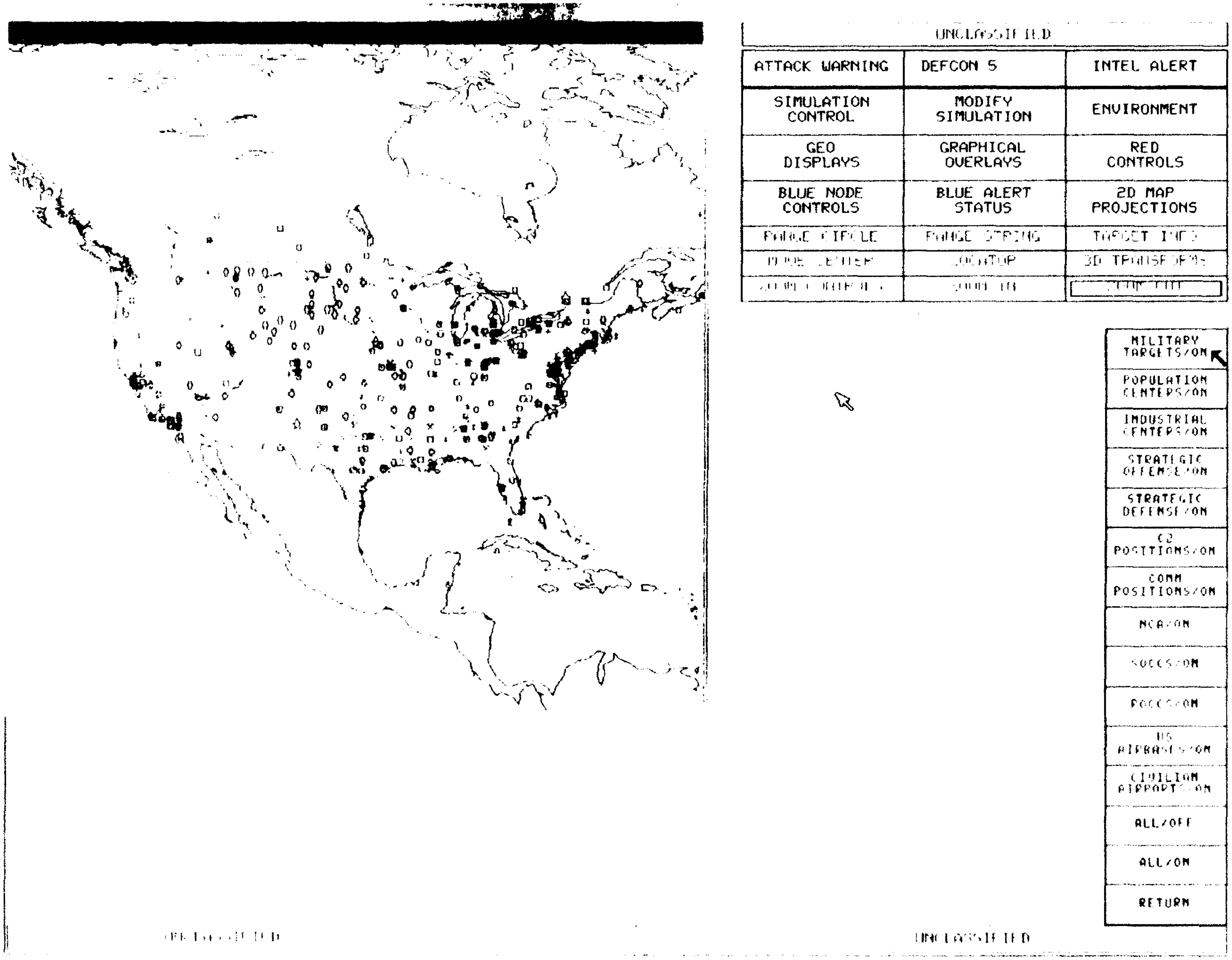
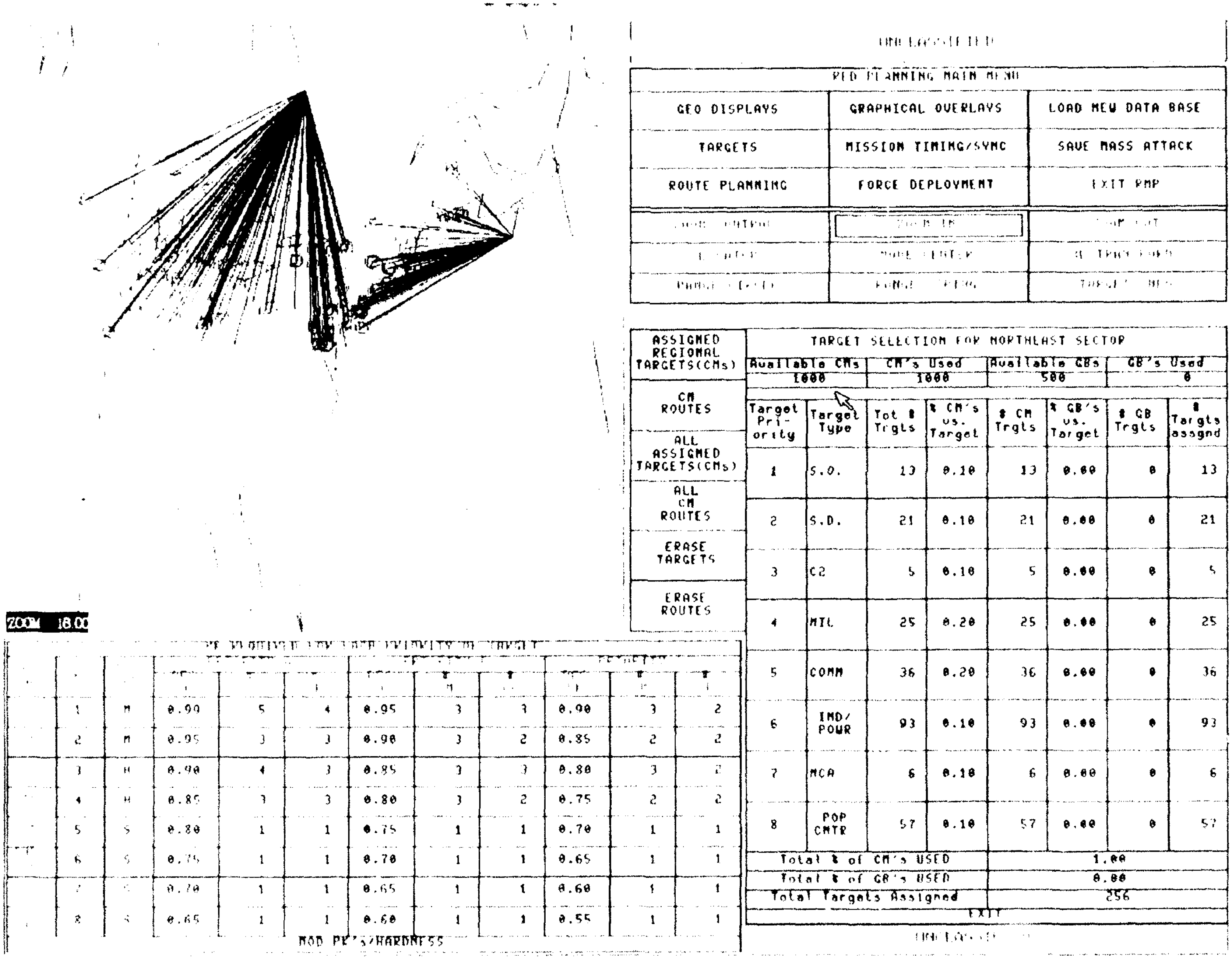

UNCLASSIFIED
ATTACK WARNING
DEFCON 5
INTEL ALERT
SIMULATION CONTROL
MODIFY SIMULATION
ENVIRONMENT
GEO DISPLAYS
GRAPHICAL OVERLAYS
RED CONTROLS
BLUE NODE CONTROLS
BLUE ALERT STATUS
2D MAP PROJECTIONS
RANGE CIRCLE
RANGE STRING
TARGET INFO
MOVE CENTER
LOCATOR
3D TRANSFORMS
ZOOM CONTROL
ZOOM IN
ZOOM OUT
MILITARY TARGETS/ON
POPULATION CENTERS/ON
INDUSTRIAL CENTERS/ON
STRATEGIC OFFENSE/ON
STRATEGIC DEFENSE/ON
C2 POSITIONS/ON
COMM POSITIONS/ON
NCA/ON
SOCCS/ON
ROCCS/ON
US AIRBASES/ON
CIVILIAN AIRPORTS/ON
ALL/OFF
ALL/ON
RETURN
UNCLASSIFIED
UNCLASSIFIED

Fig A3



UNCLASSIFIED

RED PLANNING MAIN MENU

GEO DISPLAYS	GRAPHICAL OVERLAYS	LOAD NEW DATA BASE
TARGETS	MISSION TIMING/SYNC	SAVE MASS ATTACK
ROUTE PLANNING	FORCE DEPLOYMENT	EXIT RMP
[illegible]	ZOOM IN	ZOOM OUT
[illegible]	MOVE CENTER	[illegible]
[illegible]	[illegible]	[illegible]

ASSIGNED REGIONAL TARGETS(CMs)

CM ROUTES

ALL ASSIGNED TARGETS(CMs)

ALL CM ROUTES

ERASE TARGETS

ERASE ROUTES

TARGET SELECTION FOR NORTHEAST SECTOR

Available CMs	CM's Used	Available GBs	GB's Used
1000	1000	500	0

Target Pri-ority	Target Type	Tot # Trgts	% CM's vs. Target	# CM Trgts	% GB's vs. Target	# GB Trgts	# Targts assgnd
1	S.O.	13	0.10	13	0.00	0	13
2	S.D.	21	0.10	21	0.00	0	21
3	C2	5	0.10	5	0.00	0	5
4	MIL	25	0.20	25	0.00	0	25
5	COMM	36	0.20	36	0.00	0	36
6	IND/ POWR	93	0.10	93	0.00	0	93
7	MCA	6	0.10	6	0.00	0	6
8	POP CNTR	57	0.10	57	0.00	0	57

Total % of CM's USED	1.00
Total % of GB's USED	0.00
Total Targets Assigned	256

EXIT

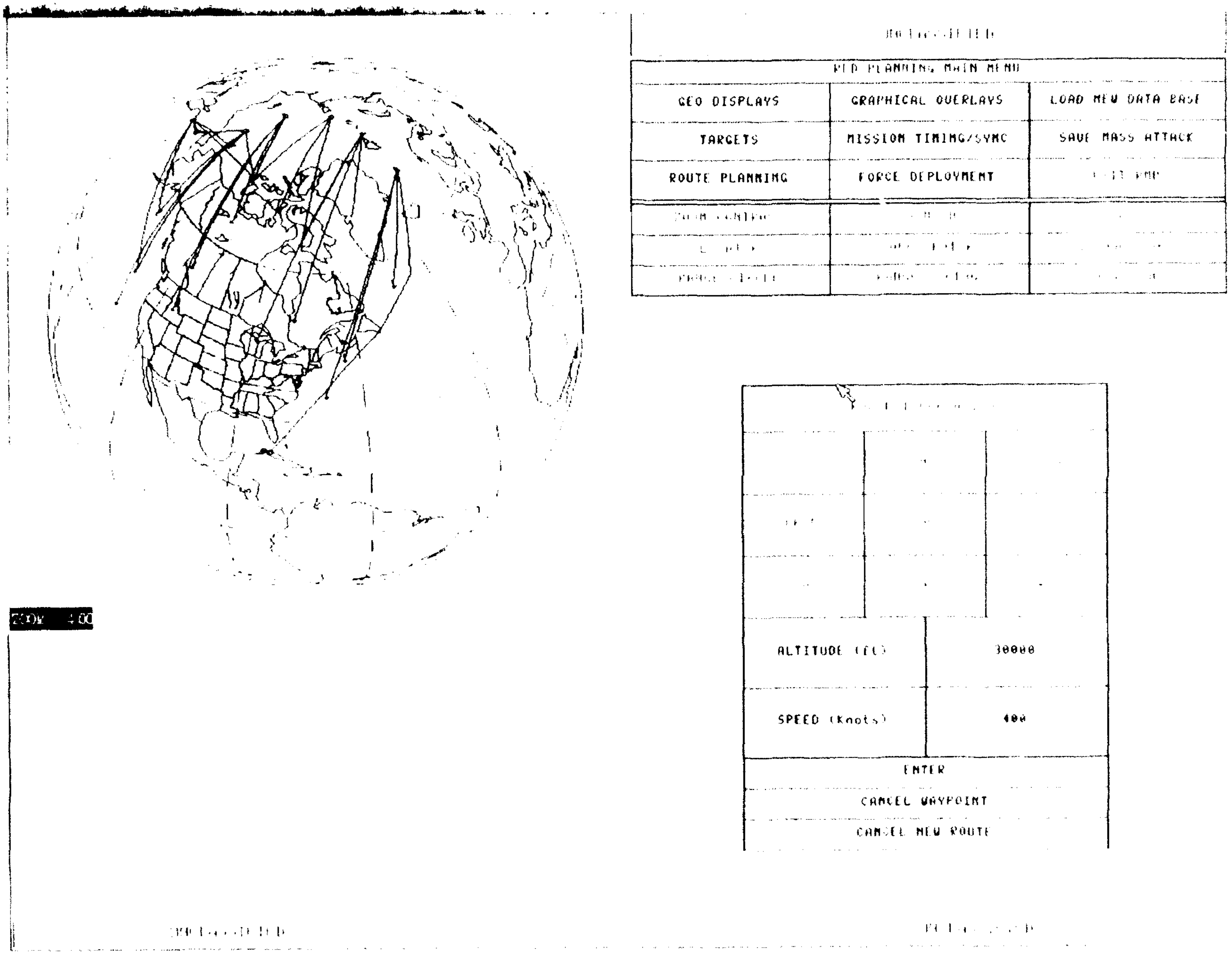
UNCLASSIFIED

ZOOM 18.00

	1	M	0.90	5	4	0.95	3	3	0.90	3	2
	2	M	0.95	3	3	0.90	3	2	0.85	2	2
	3	H	0.90	4	3	0.85	3	3	0.80	3	2
	4	H	0.85	3	3	0.80	3	2	0.75	2	2
	5	S	0.80	1	1	0.75	1	1	0.70	1	1
	6	S	0.75	1	1	0.70	1	1	0.65	1	1
	7	S	0.70	1	1	0.65	1	1	0.60	1	1
	8	S	0.65	1	1	0.60	1	1	0.55	1	1

MOD PK's/HARDNESS

Fig A4



UNCLASSIFIED

RED PLANNING MAIN MENU		
GEO DISPLAYS	GRAPHICAL OVERLAYS	LOAD NEW DATA BASE
TARGETS	MISSION TIMING/SYNC	SAVE MASS ATTACK
ROUTE PLANNING	FORCE DEPLOYMENT	[illegible]
ZOOM CENTRE	[illegible]	
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]

ALTITUDE (ft)	30000
SPEED (Knots)	400

ENTER

CANCEL WAYPOINT

CANCEL NEW ROUTE

UNCLASSIFIED

UNCLASSIFIED

Fig A5

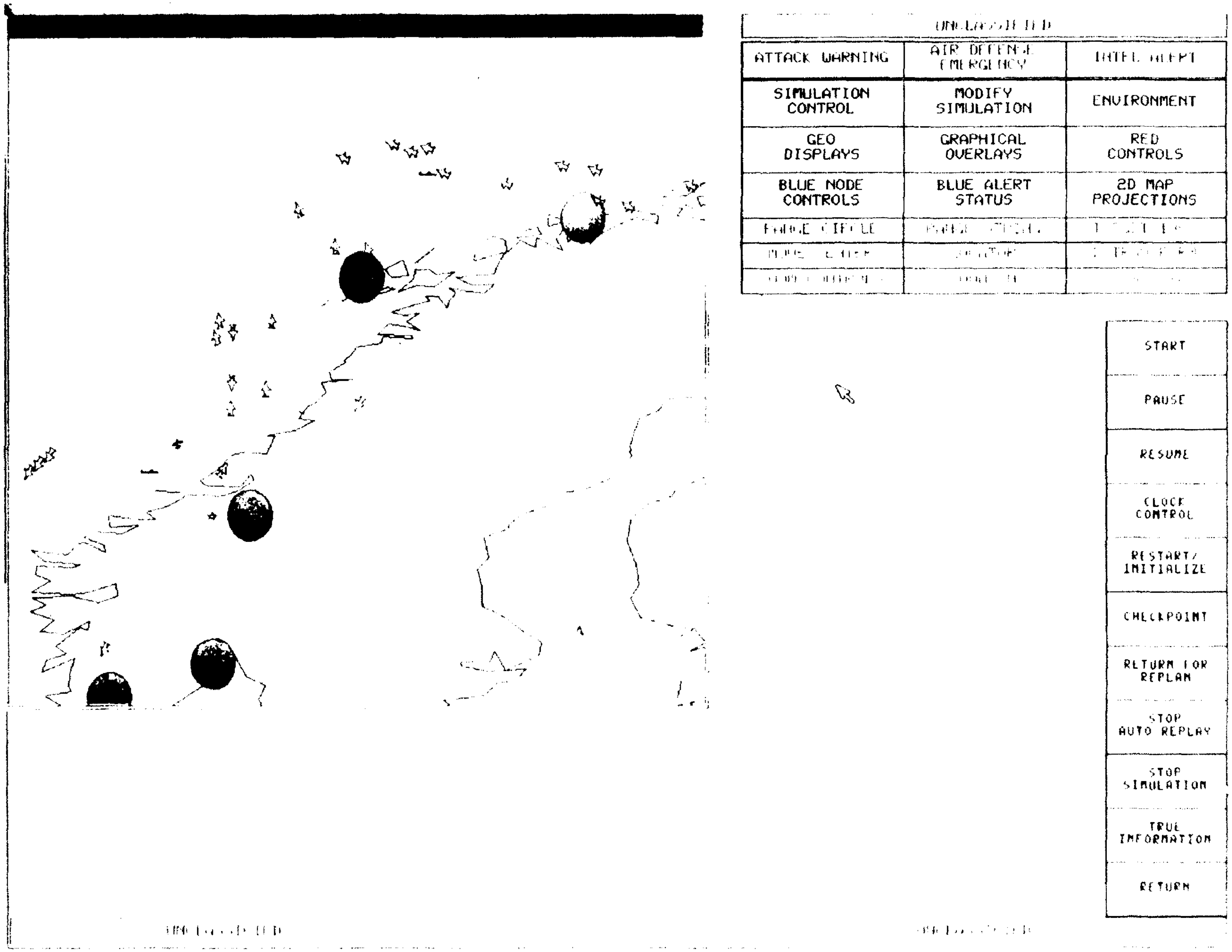

UNCLASSIFIED
ATTACK WARNING
AIR DEFENSE EMERGENCY
INTEL ALERT
SIMULATION CONTROL
MODIFY SIMULATION
ENVIRONMENT
GEO DISPLAYS
GRAPHICAL OVERLAYS
RED CONTROLS
BLUE NODE CONTROLS
BLUE ALERT STATUS
2D MAP PROJECTIONS
RANGE CIRCLE
START
PAUSE
RESUME
CLOCK CONTROL
RESTART/ INITIALIZE
CHECKPOINT
RETURN FOR REPLAN
STOP AUTO REPLAY
STOP SIMULATION
TRUE INFORMATION
RETURN
UNCLASSIFIED
UNCLASSIFIED

Fig A6

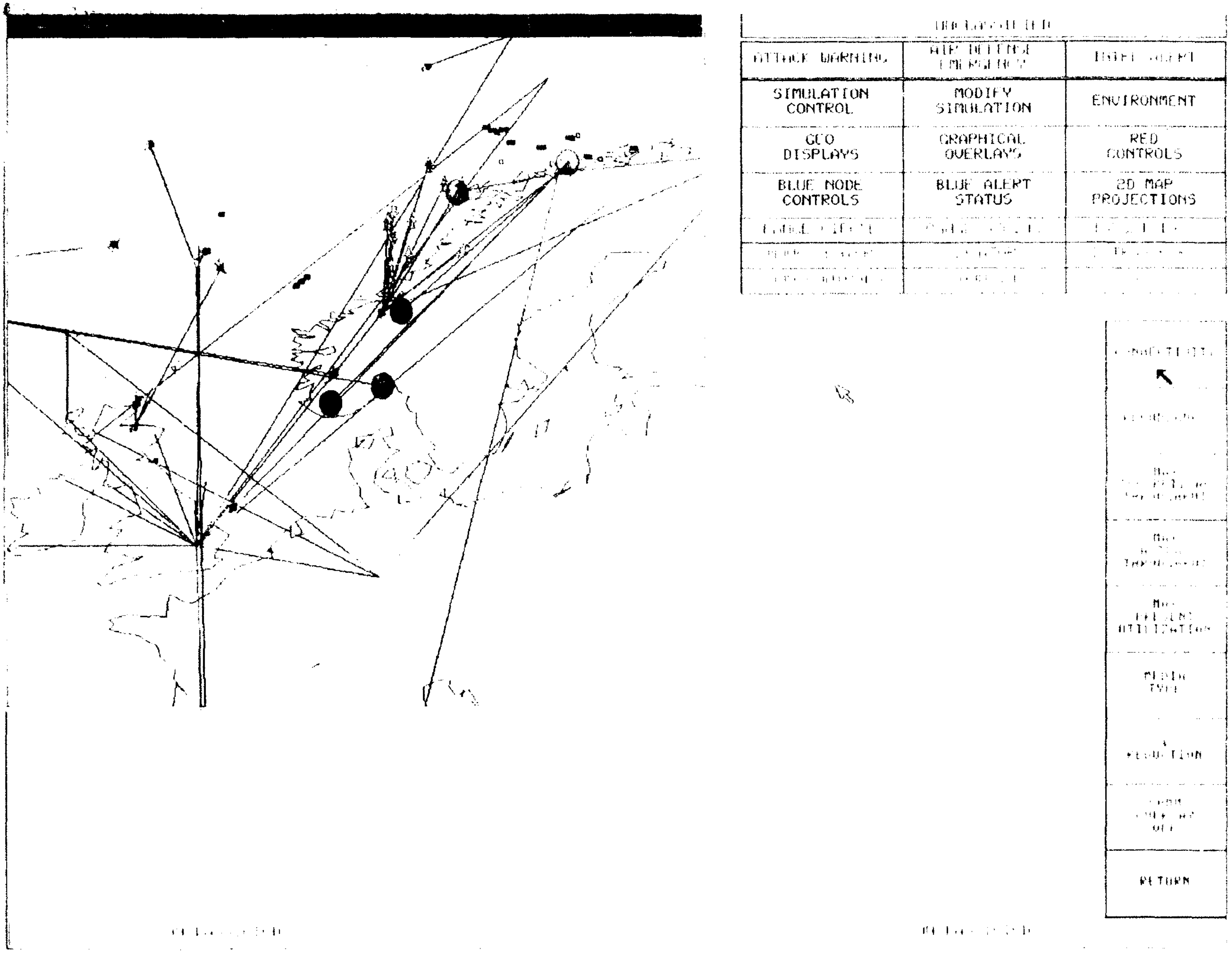
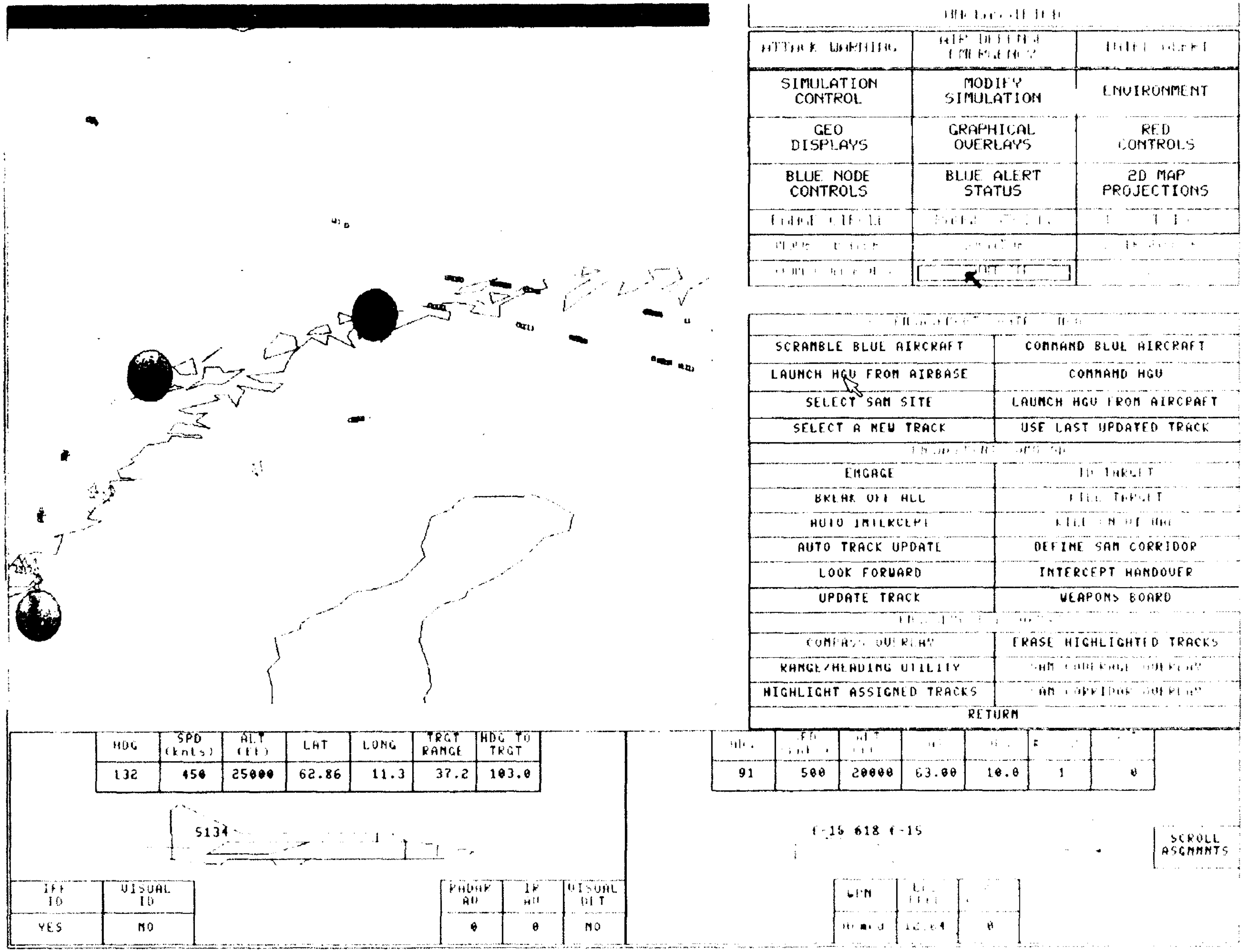

ATTACK WARNING
AIR DEFENSE EMERGENCY
SIMULATION CONTROL
MODIFY SIMULATION
ENVIRONMENT
GEO DISPLAYS
GRAPHICAL OVERLAYS
RED CONTROLS
BLUE NODE CONTROLS
BLUE ALERT STATUS
2D MAP PROJECTIONS
RESOLUTION
RETURN

Fig A7



ATTACK WARNING	AIR DEFENSE EMERGENCY	[illegible] ALERT
SIMULATION CONTROL	MODIFY SIMULATION	ENVIRONMENT
GEO DISPLAYS	GRAPHICAL OVERLAYS	RED CONTROLS
BLUE NODE CONTROLS	BLUE ALERT STATUS	2D MAP PROJECTIONS
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	

SCRAMBLE BLUE AIRCRAFT	COMMAND BLUE AIRCRAFT
LAUNCH HGU FROM AIRBASE	COMMAND HGU
SELECT SAM SITE	LAUNCH HGU FROM AIRCRAFT
SELECT A NEW TRACK	USE LAST UPDATED TRACK
ENGAGE	ID TARGET
BREAK OFF ALL	KILL TARGET
AUTO INTERCEPT	KILL [illegible]
AUTO TRACK UPDATE	DEFINE SAM CORRIDOR
LOOK FORWARD	INTERCEPT HANDOVER
UPDATE TRACK	WEAPONS BOARD
COMPASS OVERLAY	ERASE HIGHLIGHTED TRACKS
RANGE/HEADING UTILITY	SAM COVERAGE OVERLAY
HIGHLIGHT ASSIGNED TRACKS	SAM CORRIDOR OVERLAY
RETURN	

HDG	SPD (knts)	ALT (FT)	LAT	LONG	TRGT RANGE	HDG TO TRGT
132	450	25000	62.86	11.3	37.2	103.0

HDG	SPD [illegible]	ALT [illegible]	[illegible]	[illegible]	[illegible]	[illegible]
91	500	20000	63.00	10.0	1	0

S134

F-15 618 F-15

SCROLL ASGNMNTS

IFF ID	VISUAL ID
YES	NO

RADAR AQ	IR AQ	VISUAL DET
0	0	NO

WPN	[illegible]	[illegible]
[illegible]	12.64	0

Fig 18

U.S. GOVERNMENT PRINTING OFFICE

MISSION

OF

ROME LABORATORY

Rome Laboratory plans and executes an interdisciplinary program in research, development, test, and technology transition in support of Air Force Command, Control, Communications and Intelligence (C^3I) activities for all Air Force platforms. It also executes selected acquisition programs in several areas of expertise. Tecnnical and engineering support within areas of competence is provided to ESD Program Offices (POs) and other ESD elements to perform effective acquisition of C^3I systems. In addition, Rome Laboratory's technology supports other AFSC Product Divisions, the Air Force user community, and other DOD and non-DOD agencies. Rome Laboratory maintains technical competence and research programs in areas including, but not limited to, communications, command and control, battle management, intelligence information processing, computational sciences and software producibility, wide area surveillance/sensors, signal processing, solid state sciences, photonics, electromagnetic technology, superconductivity, and electronic reliability/maintainability and testability.